# Intelligent and High-Performance Behavior Design of Autonomous Systems via Learning, Optimization and Control

by

Liting Sun

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering – Mechanical Engineering

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Masayoshi Tomizuka, Chair
Professor Francesco Borrelli
Professor Claire Tomlin

Fall 2019

Intelligent and High-Performance Behavior Design of Autonomous Systems via Learning, Optimization and Control

## Abstract

Intelligent and High-Performance Behavior Design of Autonomous Systems via Learning, Optimization and Control

by

Liting Sun

Doctor of Philosophy in Engineering – Mechanical Engineering

University of California, Berkeley

Professor Masayoshi Tomizuka, Chair

Nowadays, great societal demands have rapidly boosted the development of autonomous systems that densely interact with humans in many application domains, from manufacturing to transportation and from workplaces to daily lives. The shift from isolated working environments to human-dominated space requires autonomous systems to be empowered to handle not only environmental uncertainties such as external vibrations but also interaction uncertainties arising from human behavior which is in nature probabilistic, causal but not strictly rational, internally hierarchical and socially compliant.

This dissertation is concerned with the design of intelligent and high-performance behavior of such autonomous systems, leveraging the strength from control, optimization, learning, and cognitive science. The work consists of two parts. In Part I, the problem of high-level hybrid human-machine behavior design is addressed. The goal is to achieve safe, efficient and human-like interaction with people. A framework based on the theory of mind, utility theories and imitation learning is proposed to efficiently represent and learn the complicated behavior of humans. Built upon that, machine behaviors at three different levels - the perceptual level, the reasoning level, and the action level - are designed via imitation learning, optimization, and online adaptation, allowing the system to interpret, reason and behave as human, particularly when a variety of uncertainties exist. Applications to autonomous driving are considered throughout Part I. Part II is concerned with the design of high-performance low-level individual machine behavior in the presence of model uncertainties and external disturbances. Advanced control laws based on adaptation, iterative learning and the internal structures of uncertainties/disturbances are developed to assure that the high-level interactive behaviors can be reliably executed. Applications on robot manipulators and high-precision motion systems are discussed in this part.

To my family

# Contents

# List of Figures

# List of Tables

## Acknowledgments

First and foremost, I would like to sincerely thank my advisor Professor Masayoshi Tomizuka. I am very grateful that he accepted me as a visiting student seven years ago and offered me a chance to continue as a Ph.D. student in 2015. His enthusiasm, perceptiveness and positiveness were what drew me to dive into research. It is his encouragement and valuable advice that have supported me to conquer difficulties along the path through these years and arrive where I am now. I have learned from him not only on how to conduct research, but also on how to always stay positive and humble. What I have learned from him has completely reshaped my attitude towards difficulties and will accompany and influence my entire academic life in the future.

I also would like to thank Professor Francesco Borrelli and Professor Claire Tomlin for being my dissertation committee members. Professor Borrelli opened me the door of model predictive control, which has played an important role in my research. I know Professor Tomlin by taking her course in the fall of 2015. Her rigorous logic and enthusiastic attitude have deeply touched me. I also want to thank my qualification exam committee members, Professor Roberto Horowitz, Professor Kameshwar Poolla, Professor Ruzena Bajcsy and Professor Shmuel Oren, for their support. Without the knowledge that I learned from them, I cannot finish the work in the dissertation.

Many of my recent research has been inspired and collaborated with Professor Anca Dragan. I thank her for all the inspiring and fruitful discussions. I also would like to thank Dr. Ching-Yao Chan and Professor Arnaud de La Fortelle at MINES ParisTech for their support and discussions on our research topics in “Drive for All”. They have helped me get connect to a large community focusing on autonomous driving.

Great thanks to all my mentors and senior members in the MSC lab who have helped me through my first two years in the group: Xu Chen, Wenlong Zhang, Chung-Yen Lin and Minghui Zheng. Special thanks go to Xu Chen who has offered me lots of advice and examples to learn from in terms of problem formulation, algorithm design and formal scientific writing.

I also thank all my co-authors and collaborators, especially people who have intensively discussed and helped me with the work included in the dissertation: Changliu Liu, Wei Zhan, Cheng Peng, Yongxiang Fan, Zining Wang, Yeping Hu, Jiachen Li, Hengbo Ma, Maximilian Naumann, Di Wang and Richard Lee. Without your contributions and support, I cannot finish all the researches with the current schedules. I also would like to express my thanks to the MSC lab members: Wenjie Chen, Michael Chan, Chang-Siu Evan, Xu Chen, Chi-Shen Tsai, Kan Kanjanapas, Cong Wang, Wenglong Zhang, Yizhou Wang, Raechel Tan, Minghui Zheng, Chung-Yen Lin, Junkai Lu, Yaoqiong Du, Kevin Haninger, Changliu Liu, Xiaowen Yu, Chen-Yu Chan, Shiying Zhou, Hsien-Chuang Lin, Tang Te, Yu Zhao, Shuyang Li, Wei Zhan, Cheng Peng, Yongxiang Fan, Zining Wang, Jianyu Chen, Kiwoo Shin, Saman, Fahandezhsaadi, Yujiao Cheng, Shiyu Jin, Yeping Hu, Jiachen Li, Zhuo Xu, Cheng Tang, Hengbo Ma, Jessica Leu, Changhao Wang, Xinghao Zhu, Yiyang Zhou, Lingfeng Sun, Ge Zhang, Huidong Gao, Zheng

Wu, Jinning Li, Catherine Weaver, Wu-Te Yang, Xiang Zhang and Ting Xu, for their help during my graduate study and their efforts to maintain a good research atmosphere in the group. Moreover, I also would like to thank all the visiting students in our group for your great work, particularly Sihao Chen and Chengyuan Zhang for extending the framework of safety-enhanced imitation learning on real human driving data, Xili Yi for the experimental tests of social perception algorithm, and Mu Cai for the implementation of the game policy estimation.

Great thanks also go to all my sponsors through my Ph.D. life. I thank Berkeley Deep Drive, "Drive for All", Western Digital and NSK Corporation for their funding support, and Nikon Research Corporation of America and Applied Materials for the equipment support.

I thank all the people, friends and advisors, who have encouraged me to pursue my dream through the years. Besides the people mentioned above, I would like to additionally thank Professor Chang'an Zhu in University of Science and Technology of China for his long-time support and encouragement for me to study abroad. Thank Shaoming Xiong for his suggestions and help in 2015 when I was applying for the graduate program, and Changliu Liu for her suggestions and encouragement through the years.

Finally, great thanks go to my family. Without their support and encouragement, I can never achieve this. I want to thank my parents for their endless support, patience, and love, my sisters and brother for their accompany as I grow up and their support for taking care of everything for parents while I am away from home.

# Chapter 1

# Introduction

## 1.1 Behavior of Autonomous Systems: An Overview

Humans' pursuit for automation never stops. In the past decades, automated systems have permeated and significantly impacted almost every part of our lives through the way we work, the way we live and the way we explore. For instance, in manufacturing, robotics have streamlined the factory floor and completely re-shaped the workplace, releasing human workers from tedious and dangerous tasks so that they can focus on more creative jobs. As technologies advance, more and more automated systems become autonomous. They are empowered with high-level intelligence and capabilities to make decisions without intervention from human. The emergence of autonomous systems will further boost both the safety and the efficiency of the society. For example, in manufacturing, robots do not need to sit behind the safety fence with many experts calibrating and programming them. Instead they can autonomously learn new skills and collaborate with human workers in an intelligent way. In warehouses, autonomous forklifts and guided vehicles can tremendously improve the productivity. At home, service robots can help people do houseworks and take care of children and elders, and on the road, autonomous vehicles can safely and efficiently transport goods and people for better social mobility [54].

In this dissertation, we focus on the autonomous systems in physical world, i.e., the systems with mechanical parts that perceive and act on the surrounding environment. As shown in Fig. 1.1, such autonomous systems typically need three key elements to function: the task definition module, the behavior generation module, and the perception module. The task definition module describes a set of tasks for the systems to perform. It can either be as detailed as a sequence of points in space for a machine tool to follow, or as abstract as a high-level instruction such as "clean the dishes" for a service robot to perform. The perception module, which is typically comprised of many sensors, collects information from the surrounding environment for the system. The behavior generation module, functioning as the "brain" of the system, is designed

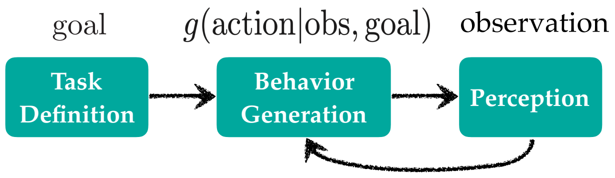


**Figure 1.1:** The three key elements of autonomous systems: the task definition module describes a set of tasks that the systems are designed to perform. It can either be as detailed as a sequence of points in space for a machine tool to follow, or as abstract as a high-level instruction such as "clean the dishes" for a service robot to perform. The perception module collects information from the surrounding environment. The behavior generation module is designed to automatically interpret the defined task and generate appropriate behaviors based on collected information in perception module.

to automatically interpret the defined tasks and generate appropriate behaviors utilizing the collected information from the perception module.

In the past decades, autonomous systems have developed rapidly, extending from *mechanical automation* to *intelligence automation*, as shown in Fig. 1.2. At the level of *mechanical automation*, our goal is to design individual machines that can complement/improve human skills in highly repetitive and unsafe tasks. Each machine is designed to fulfil a certain task in an isolated environment. We define the behavior design of such machines as *individual machine behavior design*. The main challenge hence lies in the boosting of speed, accuracy and reliability in the presence of possible uncertainties from the system models, the sensors and external vibrations. Under such circumstance, the interaction between the system and the environment is at hardware level and tends to be one-directional. Namely, the environment influences the achievable performance of the systems. As technologies in high-speed computing and artificial intelligence advance, we are now embracing a new level of automation - the *intelligence automation*. It is a level of automation that touches everyone. Intelligent systems are no longer designed to work isolated with fixed tasks, but to explore more diverse and complex tasks including interaction with human and other intelligent agents. They are entering dynamic environments full of uncertainties caused not only by non-intelligent factors but also by intelligent agents - the human beings. We define the behavior design of such intelligent systems as the *hybrid human-machine behavior design*. The challenge has therefore been extended from pursuing high performance (i.e., speed, accuracy and reliability) to empowering broader intelligence of automated machines: the ability to interpret complex tasks, the ability to understand human, and the ability to make high-level human-like decisions.

This dissertation focuses on *the design of high-performance and intelligent behavior for autonomous systems, taking into consideration of the "interaction" between the system and the environment*. We focus on two major application domains: high-precision manufacturing and autonomous driving. We address the task by answering two sets of "what" and "how":

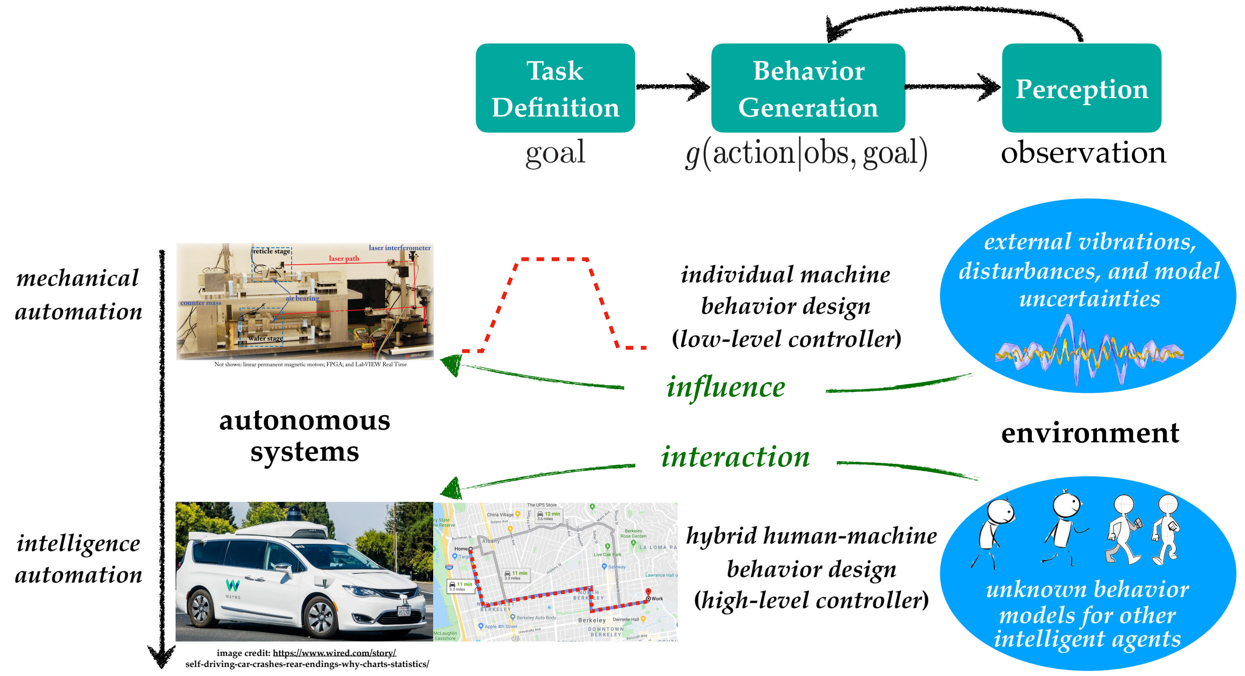

Task Definition
Behavior Generation
Perception
goal
$g(\text{action}|\text{obs}, \text{goal})$
observation
mechanical automation
intelligence automation
reticle stage
laser interferometer
laser path
counter mass
air bearing
wafer stage
Not shown: linear permanent magnetic motors; FPGA; and LabVIEW Real Time
individual machine behavior design (low-level controller)
external vibrations, disturbances, and model uncertainties
influence
autonomous systems
environment
interaction
hybrid human-machine behavior design (high-level controller)
unknown behavior models for other intelligent agents
image credit: https://www.wired.com/story/self-driving-car-crashes-rear-endings-why-charts-statistics/

**Figure 1.2:** Autonomous systems with automation extending from *mechanical automation* to *intelligence automation*. With mechanical automation, tasks are defined as reference trajectories, and the system behave to achieve high performance in the presence of external disturbances and model uncertainties. With intelligence automation, high-level tasks are given, for instance, safely and efficiently navigating from location A to location B. The system will behave in a hybrid human-machine environment.

- What characteristics of the environment should be considered for different tasks? Autonomous systems function through their interactions with the environment. The environment characteristics and the achievable performance of the systems mutually influence each other. For instance, external vibrations could directly deteriorate the precision of automated machine tools, and simultaneously the high-speed movements of the machine tools will generate new vibrations into the environment. Similarly, movements of autonomous vehicles can impact the future trajectories of other traffic participants, and vice versa. Different tasks will trigger influences from different environment factors. Precision-machine tools pay less attention to how people move around than autonomous vehicles, but worry more about the external vibrations from the ground. As behavior designers, we need to identify the key influential environment characteristics given particular tasks.

- How should the influential environment characteristics be efficiently represented? Environment characteristics are diverse and uncertain. They can be either deterministic, such as a constant noise signal, or probabilistic, such as future trajectories of a pedestrian on the road. They can also be single-mode or multi-mode. To facilitate smooth interaction between the system and the environment, we need to find a data-efficient representation of the environment and integrate it into the behavior design framework. For instance, for external vibrations, in which domain should we represent them, frequency domain or time domain? For human behavior, should we use reward functions or policies?

- What attributes should the behavior of autonomous systems possess in order to interact with the environments with particular characteristics?
  In the presence of a dynamic environment full of uncertainties, should the behavior of an autonomous system be designed to be robust, or adaptive? At which levels should such adaptability be designed? At the action level, or knowledge level or logic level as categorized in [82]? Those questions will be addressed in the following chapters in this dissertation.

- How should the behavioural attributes of autonomous systems be enabled via inter-disciplinary technologies such as control, optimization and learning?
  Attributes of the system behavior demand support from inter-disciplinary technologies. To enable adaptability, we can use adaptive control or Bayesian inference. To enable robustness, we should use robust control or stochastic optimization. To describe the behavior of human, we need to borrow available human models from other domains such as economics or cognition science. In this dissertation, we will explore appropriate tools to design intelligent and high-performance behavior for autonomous systems, spanning from problem formulation to the construction of a uniform framework.

## 1.2 Dissertation Approach

There has been a long debate about learning-based control versus model-based control. In this dissertation, we are combining the strength of the both: we exploit rich internal models regarding different environment components, and inform and constrain the learning processes by such prior knowledge. For instance, in human-machine hybrid scenarios, to describe human behavior, we utilize the structure of human decision-making models from economics and cognition science, represent it via reward functions and learn the key parameters from data. We also formulate the social factors in human-human interaction based on utility theory and learn the weight associated with them. For the high-performance individual machine behavior, we utilize internal models of the external vibrations and joint frictions, and only learn the central frequencies and model parameters to reconstruct them.

We divide this dissertation into two parts. The first part focuses on high-level intelligent *hybrid human-machine behavior* design, considering the interaction between human and autonomy. The second part discusses low-level high-performance *individual machine behavior* design, dealing with disturbances and model uncertainties to make sure that the high-level intelligent decisions/actions can be executed accurately and reliably. An overview of the approaches and emphasis of the two parts are given in Fig. 1.3.

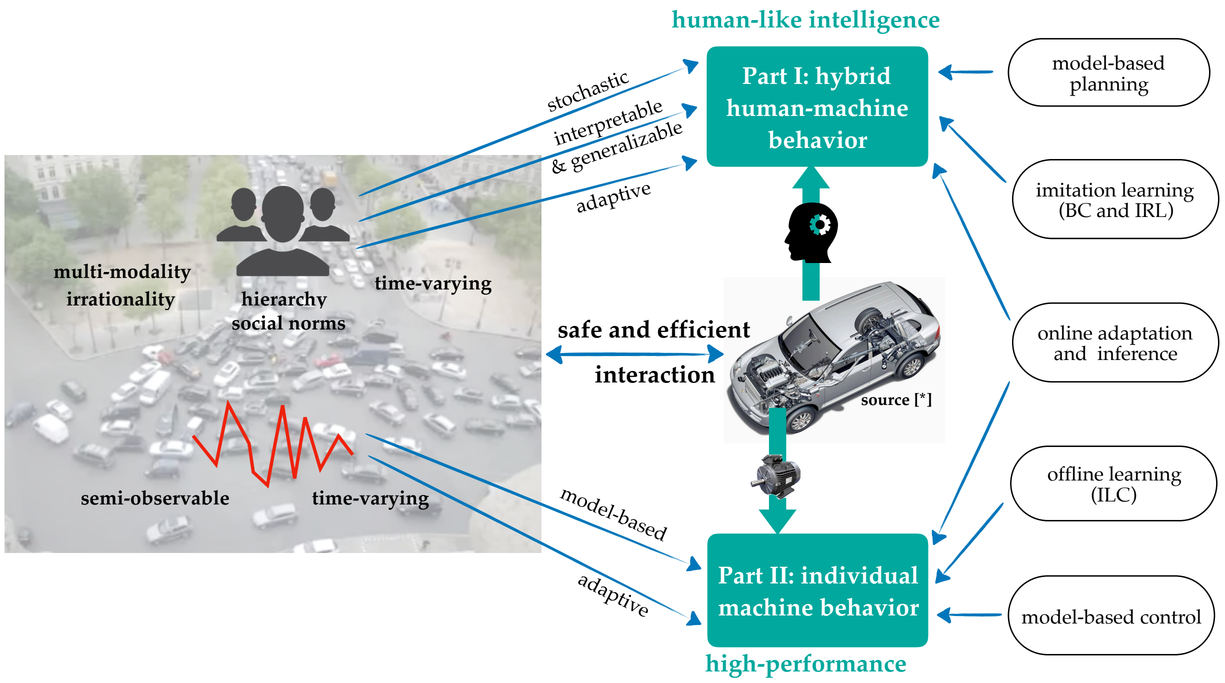


**Figure 1.3:** An overview of the challenges and approaches in designing both hybrid human-machine behavior and individual machine behavior for autonomous systems interacting with environments where both intelligent agents and non-intelligent disturbances coexist. In Part I, for *hybrid human-machine behavior*, our goal is to achieve human-like intelligence. We consider five important features of human behavior: multi-modality, irrationality, hierarchy, sociality, and time-varyingness. Correspondingly, a stochastic, interpretable and adaptive framework is formulated via model-based planning, imitation learning (both behavior cloning (BC) and inverse reinforcement learning (IRL) are included) and online adaptation. For *individual machine behavior*, the goal is to assure accurate execution of the high-level commands in the presence of unknown and time-varying disturbances and uncertainties. Hence, Part II focuses on disturbance attenuation, utilizing tools such as model-based control, offline iterative learning and online adaptation. Through the hierarchical behavior design, the autonomous system can interact with environment safely and efficiently. *Source*: https://mechanical-engg.com/gallery/image/1630-car-internal-parts.jpg*.

## 1.2.1 Hybrid human-machine behavior design

*We emphasize three key aspects, safety, efficiency and being human-like for hybrid human-machine behavior*. Towards this goal, we leverage the power from control, optimization, learning, and cognitive science. The Theory of Mind (TOM) [147] argues that human has the ability to think about mental states, both our own and those of others. Moreover, human has the ability to understand that other people's thoughts and beliefs may be different from our own and to consider the reasons behind that. We render such ability for autonomous systems by representing human behavior via structural reward/cost functions, assuming that people are maximizing their utilities under some measures, for instance, expected or non-expected. Such representation is not only data-efficient, but also intrinsically facilitates the formulation of interaction-aware behavior design, as discussed in [117]. Moreover, we integrate human-like mechanism into the structure of the reward/cost functions. Instead of assuming human as perfect rational optimizers, we exploit the following five important properties of human behavior:

- Human's decision-making process is hierarchical, including both discrete decisions and continuous optimization.
- Human behavior is probabilistic with multiple modes. In this dissertation, we refer multi-modality as multiple preferences for different people.
- Human behavior is influenced by social norms.
- Human behavior is not necessarily rational in the sense of optimizing their expected utilities. They can be systematically biased, and more appropriate measures are needed to describe such biases.
- Human behavior is time-varying. People often change mind/policies during interaction and such switch can directly influence how they behave.

By integrating the above features into the structure design of reward/cost functions of human, we have proposed a uniform framework to generate interaction-aware behaviors of autonomous systems at different levels. At the perception level, the system can infer from the behavior of other agents to compensate for the limit of its own sensors, just as humans do. At the prediction level, instead of passively inferring all possible actions of other agents, the system actively reasons what the other agents would do if I take specific actions. At the action level, the system learns to consider social norms and imitate how people behave given observations.

### 1.2.2 Individual machine behavior design

The hybrid human-machine behavior design empowers high-level intelligent actions. To assure reliable execution of such actions in the presence of model uncertainties and other external disturbances, high-performance *individual machine behavior design* is desired. For instance, for autonomous vehicles, velocity-tracking commands sent to the actuators should be accurately executed for safe maneuvers; for industrial robots, the trajectory tracking commands should be strictly followed to achieve high-precision manufacturing and assembly.

*Our approach for individual machine behavior design is to enhance model-based feedback control via offline uncertainty learning and online parameter adaptation.* As discussed above, disturbances, particularly external vibrations, are typically time-varying and environment-dependent. Moreover, since sampling rates are limited, there are external disturbances that cannot be directly observable, which makes them more difficult to attenuate. Another key feature of individual machines is the high repeatability of their tasks. For example, a wafer scanner for semiconductor manufacturing will repeat the same scanning profile for millions of times. In assembly lines, a pick-and-place robot manipulator will execute the same action without stopping. Utilizing such unique properties, we design the low-level controller via two-degree-of-freedom (2DOF) controller design. We develop selective and adaptive iterative learning controls (ILCs) to synthesize offline the feedforward control signal to handle unknown model uncertainties and invariant disturbances, combined with model-based adaptive feedback control to attenuate time-varying and even unobservable disturbances. The design of feedforward and feedback controllers are interlaced where prior model information and learning are synthesized.

## 1.3 Contributions

*The goal of this dissertation is to design intelligent and high-performance machine behaviors for autonomous systems interacting with complicated environments including intelligent agents such as human and non-intelligent influential factors such as external disturbances and model uncertainties.*

Autonomous systems function through their interactions with environment. As the working environment becomes more and more complicated - from being isolated to public spaces, and from non-intelligent factors to intelligent agents - their behaviors should empower more intelligence and adaptation to enable safe and efficient operations. This dissertation aims to address the challenges towards intelligent hybrid and high-performance individual machine behavior design by merging the strength from control, optimization, learning, and cognitive science.

More specifically, inspired by the levels of interaction (whether the influence is mutual or mainly semi-directional), we have divided this dissertation into two parts, *hybrid human-machine behavior design* in Part I for high-level intelligence and *individual machine behavior design* in Part II for high performance at the execution level. In each part, we motivate our design by identifying and constructing an efficient and interpretable representation of the unique characteristics of the environment at different levels. Part I models the stochastic (multi-modal), hierarchical, irrational, social, and time-varying human behaviors via structured reward functions in a game-theoretic setting, and Part II describes the unknown and time-varying external disturbances and model uncertainties via their internal models. Built upon that, we establish a set of methodologies in each part to equip autonomous systems corresponding capabilities to interact with such environment at both the cognition level and execution level. As shown in Fig. 1.4, for intelligent cognition, we have developed frameworks for structured reward design and learning (i.e., hierarchy, irrationality, and social compliance in reward design), and interactive behavior prediction and planning including social perception scheme, socially-compliant planning, game policy-aware motion planning and safety-enhanced behavior cloning. For high-performance execution, we have designed learning-based feedforward and adaptive feedback controllers to attenuate influences caused by both external disturbances and model uncertainties.

Contributions of this dissertation are summarized as follows.

### 1.3.1 Interaction-Aware Planning-based Human Behavior Prediction

Intelligent autonomous systems intensively interact with human. In a two-agent game-theoretic setting, the robotic system should predict the behavior of the human within the preview horizon and make decisions based on that. *We formulate the prediction to be interaction-aware* [132, 161]. Namely, instead of letting the robotic system predict what the human will do based on historical observations of both behaviors and the environments, we let it reason about what the human will do if it took specific actions under the same circumstance.

The essence of human behavior prediction is to construct an effective human behavior model [70, 157, 49]. Many such models have been proposed, spanning from rule-based ones [8, 106] to data approximation ones based on (deep) learning [104, 30, 77, 47]. Rule-based approaches can hardly capture the stochastic property of human behavior. On the other hand, the data approximation methods typically represent human behavior/policies as deep neural networks (NNs), the parameters of which are learned by directly matching the likelihood of real human demonstrations in a dataset. Such data approximation approaches can generate excellent performance on given datasets, but they suffer from poor generalizability to other scenarios and vague interpretability since it is hard to incorporate prior knowledge into such methods.

*We represent the distribution of human behavior with reward/cost functions under different measures, and translate the prediction problem as an interaction-aware planning problem*. Such representation is human-like since it aligns well with the Theory of Mind, empowering the ability of robotic systems to treat humans as intelligent agents whose decisions and actions are generated via similar logics/procedures as the robotic systems: based on the principle of maximum entropy [166], the probability of a trajectory is approximately proportional to the exponential of its reward. Moreover, as an abstract representation, it intrinsically leverages the power of model-based planning and learning. Hence, compared to rule-based methods, it can describe stochastic behaviors. Compared to data approximation approaches, it is more interpretable and generalizes better to new scenarios.

More details regarding this point will be presented in Chapter 2.

### 1.3.2 Hierarchical, Irrationality-Aware and Socially-Compliant Reward Design and Learning

To increase the expressive ability of the reward functions, their structure design plays an importance role. *We propose to design and learn hierarchical, irrationality-aware and socially-compliant reward functions [132, 136, 135], mimicking the human decision-making procedure.*

First, human behaviors are socially compliant (Chapter 3). For instance, at the action level, when a human driver wants to change lanes, it is most likely that he/she will leave a gap between himself/herself and the vehicle on the target lanes instead of cutting in abruptly. At an intersection, if a driver finds that another pedestrian or driver is waiting for him to pass first, he/she will probably accelerate more aggressively to show his/her courtesy to others instead of driving comfortably as he/she wishes. Such social compliance is in general not a directly measurable variable. We formulate and learn it from human demonstrations. Specifically, based on the game-theoretic two-agent framework in Chapter 2, *we define a courtesy term as the positive difference of human's optimal utilities between what the robot actually does and what the human expects the robot to do* [135]. Namely, the courtesy term quantifies the positive cost increase of the human given the robot's specific actions. We integrate such courtesy term as a feature in the reward function and learn its weighting from real human driving behaviors.

Second, human behaviors are hierarchical (Chapter 4). It includes both discrete decisions and continuous actions. For instance, when human drivers are changing lanes, they will first decide whether to merge in front of the other vehicle, or merge from the behind. Under each decision, the reward functions to generate the continuous maneuvers for lane-change trajectories can be different. If the driver decides to merge in front, he/she might care little about comfort but more about courtesy and speed. If the driver decides to merge from the back, he/she probably cares more about comfort but less about speed. To describe such hierarchical structure in human behavior, we develop a hierarchical inverse reinforcement learning (IRL) scheme based on Bayesian hierarchical modelling. It learns the cost functions for both discrete and continuous layers [132].

Third, human behaviors are not always rational in the sense of optimizing their expected utilities (Chapter 5). When using cost functions to represent human behaviors, there is a common assumption that human is noisily rational, as in [118]. Namely, the probability of human's behavior is approximately proportional to the exponential expected utilities of that behavior [166, 14]. There is only one parameter, $\beta$, controlling the rationality level of such model. However, a variety of evidence from other domains such as economics has shown that human's decisions are structurally biased [55], such as the framing effect, risk-seeking behavior and loss-aversion behavior, as discussed in [55, 146]. A single rationality level cannot describe the internal mechanism for these biases and thus fails to accurately predict human's behaviors. We propose to design the reward functions with a non-expected utility theory (NEUT) - the cumulative prospect theory (CPT) [146] which was developed to describe the irrationality of human behavior.

Based on that, a two-step IRL approach is formulated to learn the reward parameters from human demonstrations.

### 1.3.3 Socially-Compliant Behavior Generation

With reward functions as an efficient representation of human behavior, the next step is to generate robotic behaviors in a safe, efficient and human-like manner. We let robots generate socially-compliant behaviors at both the action level and the perception level.

At the action level, instead of being selfish, we let the robot be courteous (Chapter 3). *It minimizes not only its own cost functions but also the courtesy term, as discussed above, to encourage the robot to care about the influences of its actions to other road participants*.

At the perception level, *we let the robot learn how human deal with uncertainties, utilizing the others' behavior as signals to enhance its beliefs to the environmental states* (Chapter 6). For example, when human drivers are not sure about the speed limit, they typically follow the traffic. We consider uncertainties from both limited sensor ranges and physically unobservable group behaviors such as traffic speed and driving styles in local areas [134]. *We treat multiple agents as a distributed sensor network, and formulate the behavior planning of robotic systems as a partially observable model predictive control problem*. Namely, from the reward functions and behaviors of other agents, we can infer states of the environments and generate optimal behavior for the robots based on that.

### 1.3.4 Policy-Aware Game-Theoretic Planning

In a game-theoretic two-player setting, beyond the reward functions of both agents (human and the robot system), there is another key component influencing the way how two agents interact: the interaction policy (or game policy). Each agent, particularly the human side, can be either fully cooperative or semi-cooperative (two agents being leader-follower mode), or aggressively competitive, or completely ignoring the other side. What makes the problem more challenging is that human is time-varying, i.e., their interactive policies can change as interactions continue. Wrong interpretation of the interaction policies for human can directly lead to inaccurate behavior prediction, which might consequently generate dangerous manuveuers for the autonomous system. For instance, at an intersection, if the robot car believes that a human driver is cooperative when he/she is actually ignoring the robot car, collision can occur. On the contrary, if the robot car believes a human driver as a competitive driver when he/she actually is cooperative, the robot car might behave too conservatively and the efficiency will decrease.

*We address this challenge by formulating it as a partially observable model predictive control problem* (Chapter 7). We enumerate five different game policies: 1) ignoring, 2) constant, assuming the other agent maintaining previous actions, 3) Nash policy, 4) Stackelberg (Leader-Follower) policy, and 5) cooperative policy with a Parato equilibrium. At each step, based on the reward functions and observed actions of both agents, we update the robot's beliefs over the five different interaction policies, and generate safe interactive actions accordingly. Furthermore, we also learn the distribution of different interaction policies over different maps using real human driving data. Such an empirical study and analysis can generate a prior model/belief for human's interaction policy, facilitating reasonable assumptions for game-theoretical studies for human driving behaviors.

### 1.3.5 Safety-Enhanced Behavior Imitation

With structured reward functions for both the human and the autonomous systems, the next question is to efficiently generate actions that maximize the reward function of the robotic systems. Autonomous vehicles typically need to respond to environmental changes within 0.1 seconds, i.e., the pipeline including perception, prediction and behavior and motion planning needs to be updated at a frequency of 10Hz. Hence, it is critical that the optimal (suboptimal) actions with respect to a given reward function can be found efficiently.

There are two major approaches to generate behaviors: behavior cloning (BC) and optimization solving. While BC, as a learning-based approach, is efficient online, it suffers from un-guaranteed safety and feasibility. On the other hand, the optimization-based approach is a formal structure to systematically assure safety and feasibility enforcement, its efficiency, however, can hardly satisfy the real-time requirement, particularly for nonlinear non-convex optimization formulations.

*The approach we propose in this dissertation regarding this challenge is to hierarchically combine a long-term learning policy with an execution layer based on optimization* [133] (Chapter 8). The policy layer, represented via a neural network, is trained offline via BC. Taking a variety of features as inputs, it outputs a long-term trajectory represented by a sequence of waypoints. Hierarchically, the execution layer takes the long-term waypoints as inputs, and yields a refined short-term control action with enhanced safety and feasibility by solving either a short-term nonlinear optimization or a long-term quadratic programming (QP) problem. Moreover, to make the BC more robust, we develop an online sampled data augmentation (DAgger) to automatically re-label the test scenarios with poor performance and augment the training dataset for BC. Performance of the proposed structure on both synthetic driving data and real human driving data will be shown in Chapter 8.

### 1.3.6 Adaptive Attenuation of Disturbances Beyond Nyquist

To assure that the high-level commands generated from hybrid human-machine behaviors can be reliably executed with high performance, the low-level controller is required to be robust in the presence of external disturbances and model uncertainties. External disturbances can come from either external vibrations, or imperfection of the internal structures of the mechanical parts. For example, vibrations from the ground will excite the resonances in machine tools, and minor eccentric errors in motors will generate vibrations. Most of such external disturbances have their energy focused on several particular frequencies which, however, are commonly unknown, time-varying and environment-dependent. Moreover, since the sampling rates cannot be arbitrarily high, there are disturbances that cannot be directly recovered via measurement, i.e., the frequencies of the disturbance are beyond the Nyquist frequency[1]. Such un-observability makes the attenuation of external disturbances more difficult.

*We develop an adaptive multirate extended-state observer (MESO) to attenuate such high-frequency disturbances* [130] (Chapter 9). Compared to previous related works [12, 164, 131], no exact disturbance model is required. Alternatively, we exploit the structural information of disturbances and leverage learning to update the exact model. A two-step procedure is designed: Step I - the *disturbance estimation* step - estimates the system states as well as the disturbances via a multirate extended-state observer (MESO) based on slow-rate measurements and the current updated disturbance model; and Step II - the *model parameter update* step - updates the disturbance model parameters via recursive least-square (RLS). By iteratively repeating the two steps, the unknown beyond-Nyquist disturbances can be accurately estimated and thus compensated even though their influences to the output cannot be effectively captured by measurements. Moreover, the proposed closed-loop system can be reformulated as an add-on multirate-observer based compensator. It is compatible with any pre-designed feedback controllers such that any existing closed-loop control properties can be well preserved.

### 1.3.7 Selective and Adaptive Iterative Learning Control with Iteration-Varying Disturbances and Uncertainties

Beyond feedback control, feedforward control is also well applied to enable high-performance individual behaviors of autonomous systems. Iterative learning control (ILC) [10] is such an effective scheme to automatically learn the feedforward signal for repetitive tasks, taking into consideration potential repetitive model uncertainties and disturbances. Applications can be found in many high-precision manufacturing machines [89, 165] and autonomous vehicles such as navigation in off-road terrain[99], autonomous parking[101] and modelling of steering dynamics [57].

Even though ILC works well with repetitive tasks, model uncertainties and external disturbances, it lacks the robustness in terms of iteration-varying signals. Practically, it is unrealistic to assume that all model uncertainties and external disturbances remain the same through iterations of tasks. The resonant vibrations excited by external vibrations might start with different initial conditions at different task trials, and joint friction forces on robotic manipulators tend to change with mild speed changes at different trials. To enhance the robustness of ILC, we propose *selective and adaptive ILC methodologies which can adaptively identify the model uncertainties and disturbances and adjust the learning*

[1]Nyquist frequency, denoted as $F_N$, is half of the sampling frequency $F_s$ of the samplers, i.e., $F_N = F_s/2$.

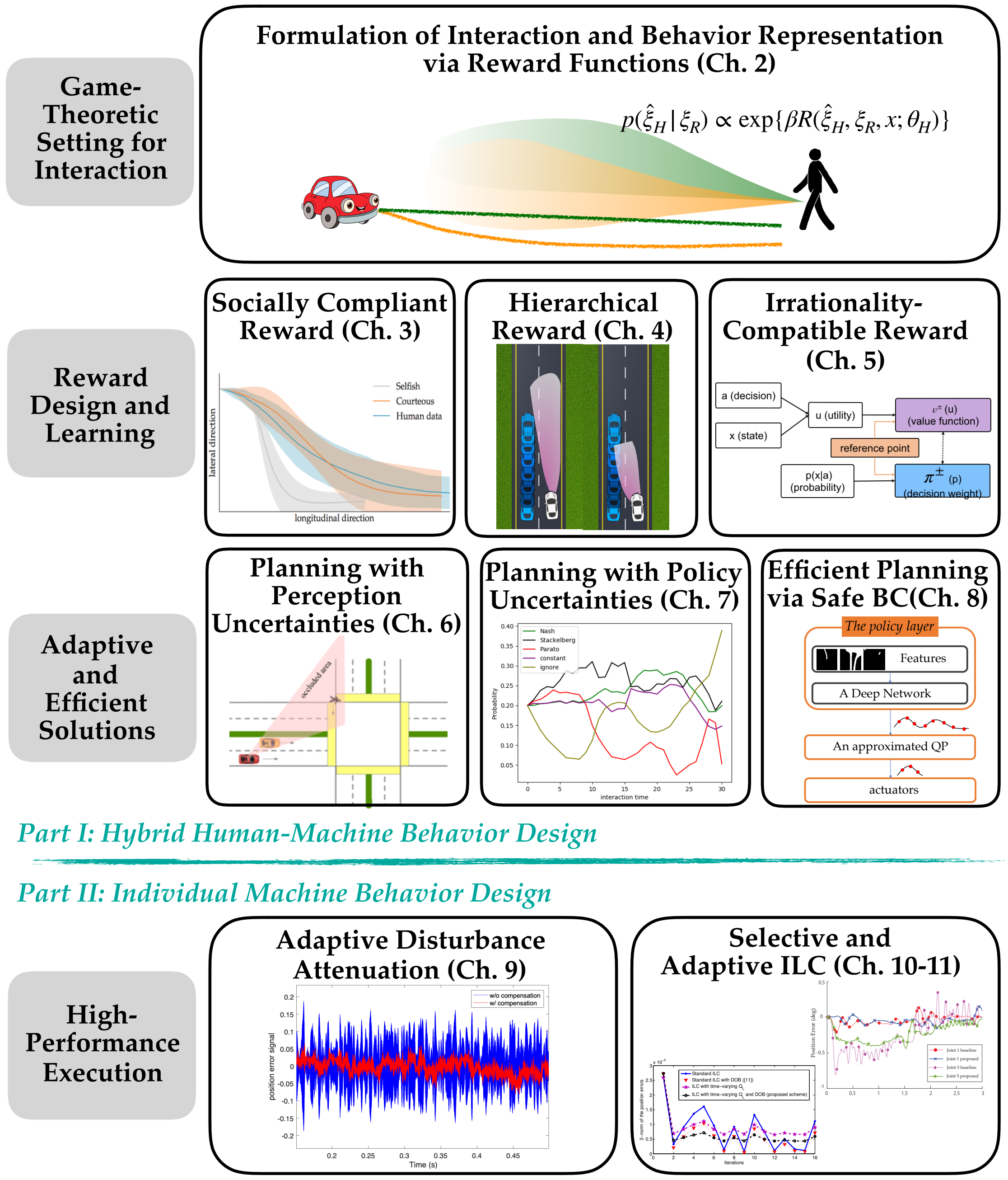

Game-Theoretic Setting for Interaction
Formulation of Interaction and Behavior Representation via Reward Functions (Ch. 2)
$p(\hat{\xi}_H | \xi_R) \propto \exp\{\beta R(\hat{\xi}_H, \xi_R, x; \theta_H)\}$
Reward Design and Learning
Socially Compliant Reward (Ch. 3)
Hierarchical Reward (Ch. 4)
Irrationality-Compatible Reward (Ch. 5)
Adaptive and Efficient Solutions
Planning with Perception Uncertainties (Ch. 6)
Planning with Policy Uncertainties (Ch. 7)
Efficient Planning via Safe BC(Ch. 8)
Part I: Hybrid Human-Machine Behavior Design
Part II: Individual Machine Behavior Design
High-Performance Execution
Adaptive Disturbance Attenuation (Ch. 9)
Selective and Adaptive ILC (Ch. 10-11)

**Figure 1.4:** Dissertation outline. Chapters 2-8 belong to Part I: Chapter 2 presents the game-theoeretic formation. Chapters 3-5 describe the reward design and learning with different emphases on social-compliance, (Ch. 3), hierarchy (Ch. 4) and irrationality (Ch. 5). Chapters 6-8 focus on adaptive and efficient solutions of the interaction-aware planning problem, i.e., planning with perception uncertainties (Ch. 6), planning with policy uncertainties (Ch. 7) and efficient planning with safety-enhanced behavior cloning (Ch. 8). Chapters 9-11 comprise Part II with Ch. 9 on adaptive feedback control, Ch. 10 on selective ILC and Ch. 11 on adaptive ILC.

*loop of ILC based on that* [129, 69] (Chapters 10-11). Flexible adaptation mechanisms are designed, through either an add-on disturbance observer [129] or an integrated two-degree-of-freedom control design [69]. Experimental results on a high-precision wafer scanner system will be provided.

## 1.4 Dissertation Outline

The remainder of the dissertation is organized as follows. Chapters 2-7 belong to Part I. Chapter 2 introduces the game-theoretic frameworks, the representation format for human behaviors as well as some preliminary concepts utilized throughout the dissertation. Chapters 3-5 focus on the reward design and learning to represent different aspects of human behavior, with Chapter 3 on social compliance, Chapter 4 on hierarchy and Chapter 5 on the irrationality. With the learned reward functions, Chapters 6-8 address the problem of efficiently generating human-like behaviors with uncertainties. Chapters 6-7 tackcle the behavior planning with perceptional and policy uncertainties using Bayesian inference, respectively. Chapter 8 constructs a framework to efficiently generate behaviors/actions optimizing reward functions in a safer and human-like manner via hierarchically combined behavior cloning and optimization. In Part II, Chapters 9-11 discuss the individual machine behavior design in the presence of external disturbances and model uncertainties. Chapter 9 focuses on adaptive attenuation of high-frequency external vibrations and Chapters 10-11 describes, respectively, selective and adaptive iterative learning control for time-varying disturbance and model uncertainties. Some of the work has been published in [129, 133, 135, 130, 132, 134, 161, 136, 47, 69].

The relationship among different chapters are summarized in Fig. 1.4.

# Part I

# Intelligent Hybrid Human-Machine Behavior Design

# Chapter 2

# Human-Robot Interaction as A Game

In this chapter, we describe the game-theoretic setting for the design of hybrid human-machine behavior of autonomous systems. Starting with a game-theoretic formulation, we represent both the human and robots behaviors via their cost/reward functions under appropriate measures. Three measures will be discussed, including the deterministic measure, the measure based on expected utility, and the measure based on cumulative prospect theory [146]. In addition, some of the preliminaries utilized throughout the dissertation will be reviewed in this chapter, including the reinforcement learning algorithm and datasets for human driving behavior.

## 2.1 Formulation of Human-Robot Interaction

In hybrid human-machine environment, we would like autonomous systems to interact with human as other humans do. Their behaviors mutually impact each other not only through the change of shared environment states, but also through the change of the other's belief towards oneself. For example, in a highway driving scenario as shown in Fig. 2.1, the lateral maneuvers of the white car will influence the blue car's accelerations not only through where it occupies, but also through how the blue car interprets these maneuvers, and vice versa.

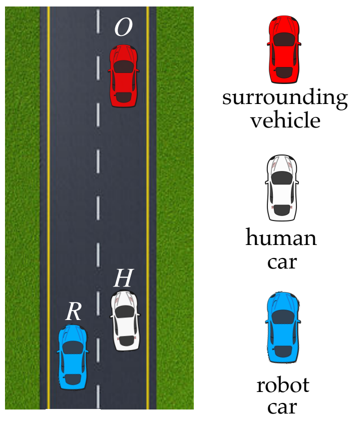


**Figure 2.1:** A example of two-play interaction game where the blue robot car interacts with the white human car. Other surrounding vehicles are treated as environment states.

Throughout this dissertation, we consider the interactive robot-human system with two agents: an autonomous system $\mathcal{R}$

and a human $\mathcal{H}$. If there are multiple autonomous robots that we control, we treat them all as a single $\mathcal{R}$. If there are multiple humans, we reason about how each of them affects the robot's behavior separately, i.e., we consider pairwise interaction. For instance, for the scenario shown in Fig. 2.1, when we model the interaction between the blue robot car and the white human car, we treat the red car as environment states. Similarly, when we focus on the interaction between the blue robot car and the red car, we treat the blue car as environment states. We denote all robot-related terms by subscript $(\cdot)_{\mathcal{R}}$ and all human-related terms by $(\cdot)_{\mathcal{H}}$.

Let $x_{\mathcal{R}}$ and $a_{\mathcal{R}}$ denote, respectively, the robot's state and control input, and $x_{\mathcal{H}}$ and $a_{\mathcal{H}}$ for the human's. Then $x=(x_{\mathcal{R}}^T, x_{\mathcal{H}}^T)^T$ represents the states of the interaction system. For each agent, we have their dynamics as

$$x_{\mathcal{R}}^{t+1} = f_{\mathcal{R}}\left(x_{\mathcal{R}}^t, a_{\mathcal{R}}^t\right), \tag{2.1}$$
$$x_{\mathcal{H}}^{t+1} = f_{\mathcal{H}}\left(x_{\mathcal{H}}^t, a_{\mathcal{H}}^t\right), \tag{2.2}$$

and the overall system dynamics are

$$x^{t+1} = f\left(x^t, a_{\mathcal{R}}^t, a_{\mathcal{H}}^t\right). \tag{2.3}$$

Meanwhile, each agent has their individual policies to generate actions based on observations and their beliefs on the other's policies. Mathematically, it can be represented by

$$a_{\mathcal{R}}^{t+1} = \pi_{\mathcal{R}}\left(x^t, \hat{\pi}_{\mathcal{H}}^t\right), \tag{2.4}$$
$$a_{\mathcal{H}}^{t+1} = \pi_{\mathcal{H}}\left(x^t, \hat{\pi}_{\mathcal{R}}^t\right). \tag{2.5}$$

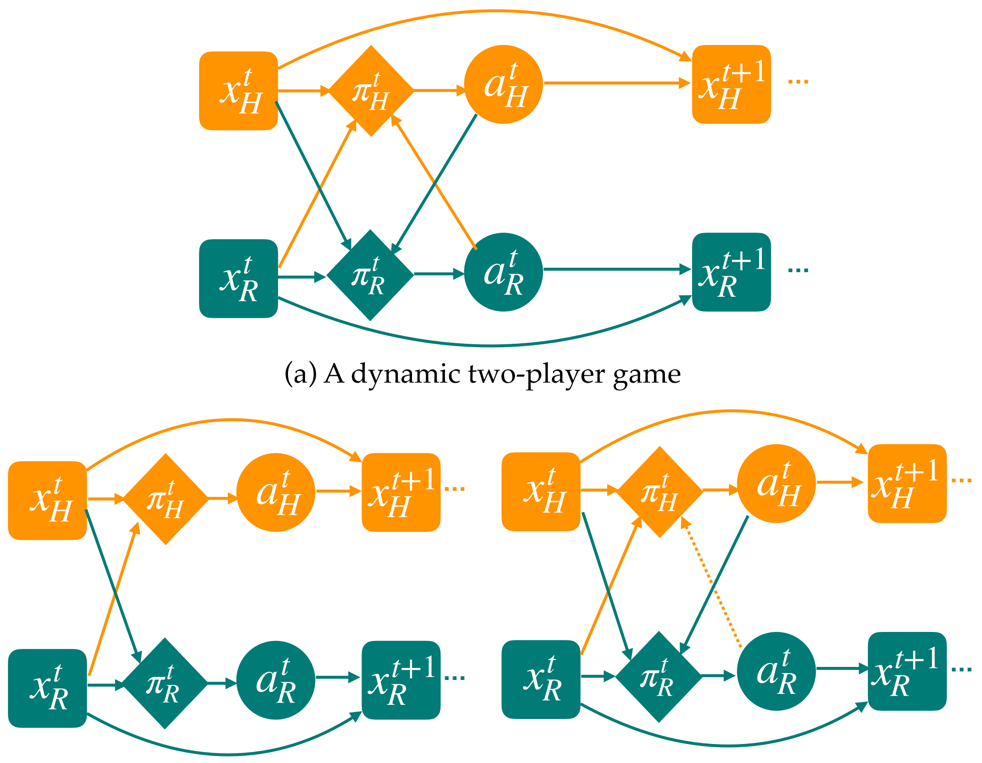
where the $(\hat{\cdot})$ represents the belief/estimate over the variable. The policies, $\pi_{\mathcal{R},\mathcal{H}}$, can be either deterministic or probabilistic.

Such interaction process can be illustrated in Fig. 2.2(a). We can see that the mutual influence between two agents has generated a loop, and practically it is not realistic to find the Nash equilibrium in real time. To make the problem solvable, two different simplifications have been developed.

## Naive simplification

As shown in Fig. 2.2(b), a naive simplification for the two-player game is to ignore the mutual influence between the future actions. Namely, the policy for each agent depends only on the states of the environment:

$$a_{\mathcal{R}}^{t+1} = \pi_{\mathcal{R}}\left(x^t\right), \tag{2.6}$$
$$a_{\mathcal{H}}^{t+1} = \pi_{\mathcal{H}}\left(x^t\right). \tag{2.7}$$



(a) A dynamic two-player game

(b) The naive simplification

(c) The interaction-aware simplification

**Figure 2.2:** Different formulations for a two-player game. (a) is the full game where the future actions for both agents mutually influence each other. (b) is a naive simplification of (a) by ignoring the interaction between the actions. (c) is an interaction-aware simplification. The interaction loop in (a) is broken by allowing the robot agent to move first as a leader.

Many techniques utilize the naive simplification. The policy can either be rule-based (i.e., assuming constant velocity as in [65, 80, 160]) or learning from data [5, 157, 123, 73]. The main problem with such simplification is that the generated robotic behaviors might be too conservative in the sense that the robot does not actively take advantage of its influence over the actions of the human.

## Interaction-aware simplification

The interaction-aware simplification tries to break the infinite interaction loop in Fig. 2.2(a) by introducing some time difference into the two agents' actions. Figure 2.2(c) shows a scenario where we let the robot propose first (the dotted orange color), and then query for potential responses from the human. Such responses are then integrate into our planners to find out our optimal behavior/policy proposals. Mathematically, (2.4) becomes

$$a_{\mathcal{H}}^{t,'} = \pi_{\mathcal{H}}\left(x^t, \hat{u}_{\mathcal{R}}^t\right), \tag{2.8}$$

$$a_{\mathcal{R}}^{t+1} = \pi_{\mathcal{R}}\left(x^t, a_{\mathcal{H}}^{t,'}\right), \tag{2.9}$$

which means that $a_{\mathcal{R}}^{t+1} = \pi_{\mathcal{R}}\left(x^t, \pi_{\mathcal{H}}\left(x^t, \hat{u}_{\mathcal{R}}^t\right)\right)$. The interaction-aware simplification, compared to the naive one, explicitly models the mutual influences at the action level, and thus can help the robots generate more aggressive behaviors, as mentioned in [118, 117, 135, 13, 78] and [52].

In this dissertation, we select the interaction-aware simplified game as our framework for hybrid human-machine behavior design.

## 2.2 Behavior Representation via Cost/Utility

### Human Behavior Representation

As discussed above, to design the robotic behavior in interaction games, we need to know the behavior generation policy of human, i.e., $\pi_{\mathcal{H}}$.

Many approaches have been proposed to represent/predict human behavior. They can be as simple as manually designed rules, i.e., assuming the person will maintain their current actions for a period of time [65, 80, 160], or as complicated as learning a human's policy via neural networks [5, 157, 123].

Meanwhile, it is a general idea in cognitive science that humans make decisions or take actions by maximizing some measure of costs/utilities under uncertainty [39], and the computation is limited, known as bounded rationality [122]. Hence, we assume humans as finite-horizon noisily rational planners. We use the framework of model predictive control (MPC) [19] (also known as receding horizon control (RHC)) to approximate the human behavior generation process.

Suppose that $C_{\mathcal{H}}$ is some measure of expected cost of the human over a horizon with length N [1]:

$$C_{\mathcal{H}}\left(x^t, \mathbf{a}_{\mathcal{R}}, \mathbf{a}_{\mathcal{H}}, x_{env}^t; \theta_{\mathcal{H}}\right) = \mu\left[\sum_{k=0}^{N-1} c_{\mathcal{H}}\left(x^{t,k}, a_{\mathcal{R}}^k, a_{\mathcal{H}}^k, x_{env}^{t,k}; \theta_{\mathcal{H}}\right)\right] \tag{2.10}$$

where $\mathbf{a}_i$=$(a_i^0, a_i^1, \cdots, a_i^{N-1})^T$ are the sequences of control actions of the robot ($i$=$\mathcal{R}$) and the human ($i$=$\mathcal{H}$) over the horizon. $x^{t,k}$ and $x_{env}^{t,k}$ with $k$=$0, 1, \cdots, N-1$ are, respectively, the corresponding sequences of human-robot system states and other environment states. Note that the environment states can include other humans/robots which

[1] In this dissertation, we denote utility as $U$ and cost as $C$. Typically, we set $U = -C$ with $C \geqslant 0$. Hence, policies that maximizes utilities are equivalent to policies that minimize costs.

are not closely interacting with the human-robot system, external disturbances, and so on. $\theta_{\mathcal{H}}$ represents the preference of the human. Hence, $c_{\mathcal{H}}\left(x^{t,k}, a^k_{\mathcal{R}}, a^k_{\mathcal{H}}, x^{t,k}_{env}; \theta_{\mathcal{H}}\right)$ represents the objective cost for state-action pairs at each step. The function $\mu : \mathcal{R}^+ \to \mathcal{R}^+$ is a measure that maps an objective cost term $c_{\mathcal{H}}$ into a decision cost which is utilized for decision/behavior generation. More details regarding the measures will be provided in Section 2.3.

Based on the principle of Maximum Entropy [166], at every time step t, the human generates a sequence of actions with a probability which is approximately proportional to the exponential negative cost. Namely,

$$P(\mathbf{a}_{\mathcal{H}}, \theta_{\mathcal{H}} | x^t, \mathbf{a}_{\mathcal{R}}, \boldsymbol{x}^t_{env}) \propto \exp^{-\beta C_{\mathcal{H}}\left(x^t, \mathbf{a}_{\mathcal{R}}, \mathbf{a}_{\mathcal{H}}, \boldsymbol{x}^t_{env}; \theta_{\mathcal{H}}\right)}. \tag{2.11}$$

where $\beta$ is a hyper-parameter that controls the rational level of the human. Only the first action will be executed, and the human will re-plan via the same process at the next time step $t+1$.

Hence, the expected cost measure $C_{\mathcal{H}}$ has defined a probability distribution over the human's future actions given observations $x^t, x^t_{env}$ and the potential actions of the robot $\mathbf{a}_{\mathcal{R}}$, which means that (2.11) represents a probabilistic policy $\pi_{\mathcal{H}}$ as in (2.8). If a deterministic policy $\pi_{\mathcal{H}}$ is preferred, we can change the distribution to a fixed policy such as a minimizer:

$$\mathbf{a}^*_{\mathcal{H}} = \arg\min_{\mathbf{a}_{\mathcal{H}}} C_{\mathcal{H}}\left(x^t, \mathbf{a}_{\mathcal{R}}, \mathbf{a}_{\mathcal{H}}, \boldsymbol{x}^t_{env}; \theta_{\mathcal{H}}\right). \tag{2.12}$$

Therefore, given appropriate measures of their expected costs, the human behavior generation policy $\pi_{\mathcal{H}}$ can be represented as

$$\pi_{\mathcal{H}}(x^t, \mathbf{a}_{\mathcal{R}}, \boldsymbol{x}^t_{env}; \theta_{\mathcal{H}}) = \begin{cases} P\left(\mathbf{a}_{\mathcal{H}} | x^t, \mathbf{a}_{\mathcal{R}}, \boldsymbol{x}^t_{env}, \theta_{\mathcal{H}}\right), & \text{probabilistic policy} \\ \arg\min_{\mathbf{a}_{\mathcal{H}}} C_{\mathcal{H}}\left(x^t, \mathbf{a}_{\mathcal{R}}, \mathbf{a}_{\mathcal{H}}, \boldsymbol{x}^t_{env}; \theta_{\mathcal{H}}\right), & \text{deterministic policy.} \end{cases} \tag{2.13}$$

## Robot Behavior Generation Framework

To design human-like behavior for the autonomous systems, based on the Theory of Mind [147], we let the robot follow the same MPC framework as human. Namely, at each step t, the robot generates its actions by minimizing some measure of its cost defined over a finite horizon N, i.e.,

$$\begin{aligned} \mathbf{a}^*_{\mathcal{R}} &= \arg\min_{\mathbf{a}_{\mathcal{R}}} C_{\mathcal{R}}\left(x^t, \mathbf{a}_{\mathcal{R}}, \mathbf{a}_{\mathcal{H}}, \boldsymbol{x}^t_{env}; \theta_{\mathcal{R}}\right) \\ &= \arg\min_{\mathbf{a}_{\mathcal{R}}} C_{\mathcal{R}}\left(x^t, \mathbf{a}_{\mathcal{R}}, \pi_{\mathcal{H}}(x^t, \mathbf{a}_{\mathcal{R}}, \boldsymbol{x}^t_{env}; \theta_{\mathcal{H}}), \boldsymbol{x}^t_{env}; \theta_{\mathcal{R}}\right) \end{aligned} \tag{2.14}$$

where $C_{\mathcal{R}}$ is defined in a similar way as $C_{\mathcal{H}}$ in (2.10). With only the first action executed, the process repeats at the next time step for re-planning.

## 2.3 Cost/Utility Measures

Different measures of the expected costs/utilities can lead to quite different behavior policies. In this dissertation, we explore three different measures, i.e., the deterministic measure, the expected-utility based measure and a non-expected utility based masure, and their applications in specific scenarios.

### 2.3.1 The Deterministic Measure

The deterministic measure ignores the uncertainties. It assumes that all the states and actions are deterministic, and the decision cost equals to the objective cost. Hence, (2.10) reduces to

$$C_{\mathcal{H}}\left(x^t, \mathbf{a}_{\mathcal{R}}, \mathbf{a}_{\mathcal{H}}, \boldsymbol{x}^t_{env}; \theta_{\mathcal{H}}\right) = \sum_{k=0}^{N-1} c_{\mathcal{H}}\left(x^{t,k}, a^k_{\mathcal{R}}, a^k_{\mathcal{H}}, x^{t,k}_{env}; \theta_{\mathcal{H}}\right). \tag{2.15}$$

With the deterministic measure, the decision cost $C_{\mathcal{H}}$ can be fully represented via $c_{\mathcal{H}}$ which can be directly learned from human demonstrations via Inverse Optimal Control (IOC) [74]. Details can be found in Section 2.5. Such a measure is applicable when uncertainties are small. Hence, it will be utilized in this dissertation to describe relatively deterministic and low-level behaviors of human, for instance, how a human finishes a turning when his/her intention has been clearly demonstrated.

### 2.3.2 The Measure based on Expected Utility Theory

In most application scenarios, human make decisions under uncertainties, and a deterministic measure cannot represent such behaviors. Hence, we explore another measure based on the expected utility theory (EUT) [114]. First introduced by Bernoulli in 1738, EUT has become the most widely adopted theory to describe how human make decisions with uncertainty. Mathematically, the process can be modelled as follows.

Compared to the deterministic measure which assumes that a state sequence $\boldsymbol{x} = [x^0, x^1, \cdots, x^{N-1}]$ is fully determined by the action sequence $\boldsymbol{a}$, EUT considers the probability of each state sequence. Define all possible state sequences under the action sequence $\boldsymbol{a}$ as a set represented by $\{\boldsymbol{x}\}$=$\{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_m\}$ for j=1, $\cdots$, m. Note that the set can contain infinite elements, i.e., $m \to \infty$, when the state sequence satisfies a continuous distribution. Let $p_j = p(\boldsymbol{x}_j)$ be the probability (or the probability density function for continuous random variables) of each state sequence $\boldsymbol{x}_j$ satisfying $\sum_j p(\boldsymbol{x}_j)$=1. Define $u(x^{k+1}_j, a^k)$ as the objective utility assigned to each pair of state $x^{k+1}_j$ and action $a^k$ at time $k = 0, 1, \cdots, N-1$ over the horizon. Note that we have $u(x^{k+1}_j, a^k) = -c(x^{k+1}_j, a^k)$.

Then, under decision $\boldsymbol{a}$, the possible outcome profile (i.e., the prospect) can be represented by $\boldsymbol{P}(\boldsymbol{a})$= $(\boldsymbol{u}(\boldsymbol{a}), \boldsymbol{p})$ where $\boldsymbol{u}(\boldsymbol{a})$=$[u(x^1_j, a^0), u(x^2_j, a^1), \cdots, u(x^N_{i,m}, a^{N-1})]^T$ is the utility vector defined on the possible state set $\{\boldsymbol{x}\}$, and $\boldsymbol{p}=[p_1, p_2, \cdots, p_m]^T$ is the corresponding probability vector (or the probability density function vector) of $\boldsymbol{x}_j$. The expected utility U of $\boldsymbol{a}$ are defined as

$$U(\boldsymbol{a}) = U(\boldsymbol{P}(\boldsymbol{a})) = \mathbb{E}_{[x^k]_j \sim \boldsymbol{p}_j}\left[\sum_{k=0}^{N-1} u(x_j^{k+1}, a^k)\right] = \sum_{j=1}^{m} p_j \sum_{k=0}^{N-1} u(x_j^{k+1}, a^k). \tag{2.16}$$

For the human and robot behavior in the two-player game, we consider uncertainties from two major sources: 1) the state uncertainties including both the environment states $x_{env}$ and the human-robot system states $x$, and 2) the uncertainties on the potential behavior of the other agent, i.e., $\boldsymbol{a}_{\mathcal{R}}$ included in the human cost and $\boldsymbol{a}_{\mathcal{H}}$ in the robot's cost. Hence, the measure function on the right-hand side of (2.10) can be specified as

$$\begin{aligned} &\mu\left[\sum_{k=0}^{N-1} c_{\mathcal{H}}\left(x^{t,k+1}, a_{\mathcal{R}}^k, a_{\mathcal{H}}^k, x_{env}^{t,k+1}; \theta_{\mathcal{H}}\right)\right] \\ = \; &\mathbb{E}_{\boldsymbol{x}\sim p_x, \boldsymbol{x}_{env}\sim p_{x_{env}}, \boldsymbol{a}_{\mathcal{R}}\sim p_{a_{\mathcal{R}}}}\left[\sum_{k=0}^{N-1} c_{\mathcal{H}}\left(x^{t,k}, a_{\mathcal{R}}^k, a_{\mathcal{H}}^k, x_{env}^{t,k}; \theta_{\mathcal{H}}\right)\right] \end{aligned} \tag{2.17}$$

where $p_x$, $p_{x_{env}}$ and $p_{a_{\mathcal{R}}}$ represent, respectively, the distribution of the state sequence $[x^{t,1}, x^{t,2}, \cdots, x^{t,N}]$ given action sequence $\boldsymbol{a}_{\mathcal{H}}$ and initial state $x^{t,0}$, the distribution of the state sequence $[x_{env}^{t,1}, x_{env}^{t,2}, \cdots, x_{env}^{t,N}]$, and the distribution of the estimate for $\boldsymbol{a}_{\mathcal{R}}$.

Hence, to learn the behavior of human based on the measure of expected utility, we need to learn not only the objective cost function as in the deterministic measure, but also the probability distributions for the states and the human's estimate/blief of the robot's future actions. We will discuss more about this point in Section 2.4.

### 2.3.3 The Measure based on Cumulative Prospect Theory

Although EUT has been adopted in many application domains as the dominant model to describe how individuals make decisions under uncertainties, there have been substantial evidences showing that human behavior often violates the EUT hypothesis in a systematic way such as framing effect, loss aversion, risk seeking and nonlinear preferences [146, 7].

Many non-expected utility theories (NEUT) were developed to explain the above-mentioned behaviors which deviate from EUT. Among them, the cumulative prospect theory (CPT), proposed by Kahneman and Tversky [146], is one study that formulates many such biased or irrational human behaviors in a uniform way. Compared to EUT in (2.16), CPT introduced two additional concepts in the definition of prospect $\boldsymbol{P}$: a value function $v$ defined on the utility and a decision weight function $\pi$ defined on the cumulative probability. Instead of using the expected utility to evaluate actions, CPT evaluate actions via a cumulative function defined as

$$\begin{aligned} V(\boldsymbol{a}) &= V(\boldsymbol{P}(\boldsymbol{a})) \\ &= \sum_{j=1}^{m} v\left(u^{+}(x_j,\boldsymbol{a})\right)\pi_j^{+} + v\left(u^{-}(x_j,\boldsymbol{a})\right)\pi_j^{-}, \end{aligned} \tag{2.18}$$

where the function $v : \mathcal{R}\to\mathcal{R}$ is a strictly increasing function, and $u^{+}(\cdot)$ and $u^{-}(\cdot)$ represent, respectively, the gains and losses of $u(\cdot)$ compared to a reference utility $u_0$. The decision weights are defined as

$$\pi_m^{+} = w^{+}\left(p(x_m)\right), \quad \pi_m^{-} = w^{-}\left(p(x_m)\right), \tag{2.19}$$

$$\pi_j^{+} = w^{+}\left(\sum_{k=j}^{m} p(x_k)\right) - w^{+}\left(\sum_{k=j+1}^{m} p(x_k)\right), \tag{2.20}$$

$$\pi_j^{-} = w^{-}\left(\sum_{k=j}^{m} p(x_k)\right) - w^{-}\left(\sum_{k=j+1}^{m} p(x_k)\right), \quad \forall j=1,\cdots,m-1 \tag{2.21}$$

where $w^{\pm}:[0,1]\to[0,1]$ are both strictly increasing functions with $w^{+}(0)=w^{-}(0)=0$, and $w^{+}(1)=w^{-}(1)=1$.

Typically, the value function $v(u)$ is convex when $u\geqslant u_0$ (gains) and concave when $u<u_0$ (losses), and it is steeper for losses than for gains. Figure 2.3(a) shows one example of the value function when $u_0=0$ is set as the reference utility. Many experiment studies have showed that representative functional forms for $v$ and $w$ can be written as

$$v(u) = \begin{cases} (u-u_0)^{\alpha}, & \text{if } u \geqslant u_0 \\ -\lambda(u_0-u)^{\beta}, & \text{if } u < u_0 \end{cases} \tag{2.22}$$

$$w^{+}(p) = \frac{p^{\gamma}}{(p^{\gamma}+(1-p)^{\gamma})^{1/\gamma}}, \tag{2.23}$$

$$w^{-}(p) = \frac{p^{\delta}}{(p^{\delta}+(1-p)^{\delta})^{1/\delta}}. \tag{2.24}$$

respectively, with $\alpha,\beta,\gamma,\delta\in(0,1]$ and $\lambda\geqslant 1$. As shown in Fig. 2.3(b), the decision weighting functions can describe the well-observed human behaviors that humans tend to over-estimate the occurrence of low-probability events but under-estimate that of the high-probability ones.

Hence, based on CPT, the cost measure in (2.10) will be specified as (the variables in

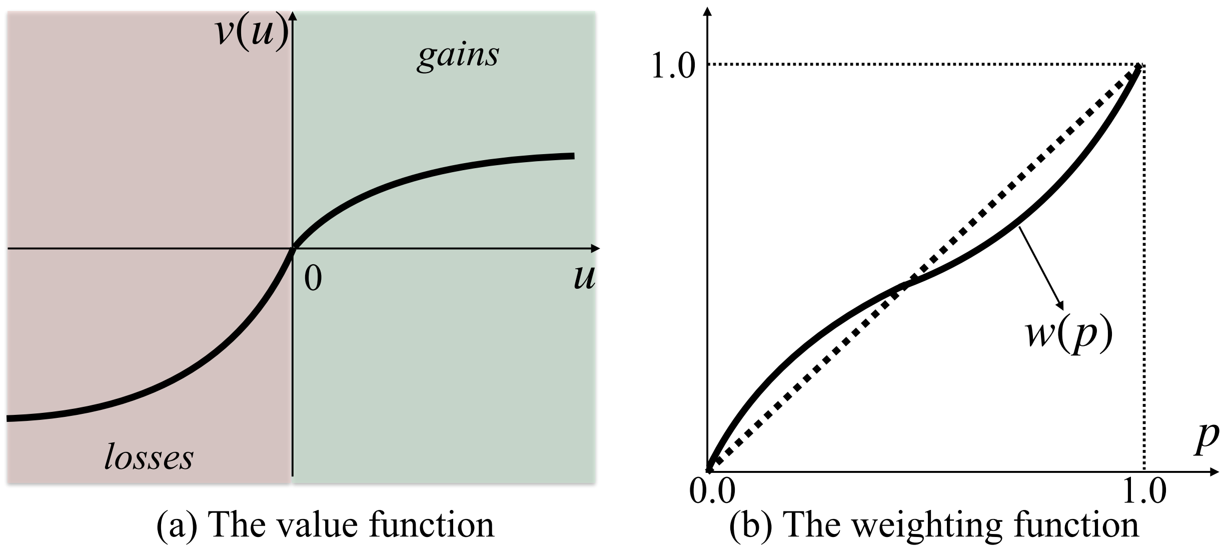


**Figure 2.3:** Examples of the value function and weighting function: (a) the value function $v$ maps the objective utility into a value for decisions, and it is convex for losses and concave for gains. (b) shows the decision weighting function $\pi$ which maps the objective probability into the decision weight. It shows that humans tend to over-estimate the occurrence of low-probability events but under-estimate that of the high-probability ones.

the second line of (2.25) will be emitted due to space limit)

$$
\begin{aligned}
&\mu\left[\sum_{k=0}^{N-1} c_{\mathcal{H}}\left(x^{t,k+1}, a_{\mathcal{R}}^{k}, a_{\mathcal{H}}^{k}, x_{env}^{t,k+1}; \theta_{\mathcal{H}}\right)\right] \\
=\ &\mathbb{E}_{\boldsymbol{x}\sim p_x, \boldsymbol{x}_{env}\sim p_{x_{env}}, \boldsymbol{a}_{\mathcal{R}}\sim p_{a_{\mathcal{R}}}}\left[\sum_{k=0}^{N-1} v^{+}\left(c_{\mathcal{H}}\right)\pi^{+}(p_x, p_{x_{env}}, p_{a_{\mathcal{R}}}) + v^{-}\left(c_{\mathcal{H}}\right)\pi^{-}(p_x, p_{x_{env}}, p_{a_{\mathcal{R}}})\right].
\end{aligned}
\tag{2.25}
$$

Therefore, we can see that beyond the objective cost and uncertain distributions utilized in EUT based measure, using the CPT measure to describe the human behavior needs to learn two more key elements: the value function and decision weighting function.

### 2.3.4 A Summary of the Measures

A summary of the three different measures is listed in Table 2.1. We can see that among the deterministic, EUT-based and CPT-based measures, CPT-based measure provides the most flexibilities to describe more diverse behavior of human. In the meanwhile, it involves more unknowns in terms of parameters, and needs more powerful learning framework to learn those unknowns from human demonstrations.

| Measures | Property | Factors to Learn |
|---|---|---|
| Deterministic measure | no uncertainties | objective cost function |
| EUT measure | with uncertainties | objective cost function<br>probability distribution over uncertainties |
| CPT measure | irrationality-compatible | objective cost function<br>probability distribution over uncertainties<br>value function<br>decision weight function |

**Table 2.1:** A summary of different measures of cost

To represent human behavior, we explore all three different measures and corresponding learning algorithms. Regarding to the robot behavior generation, we would like to design a rational robot system. Hence, *to generate the robot policies, we only utilize the first two measures in the formulation of* $C_{\mathcal{R}}$ *defined in (2.14): the deterministic measure when uncertainty can be ignored and the EUT-based measure when uncertainty is significant.*

Hence, in the following two sections, we will address two related techniques utilized to specify $C_{\mathcal{R}}$, i.e., the formulation for predicting future human actions $\boldsymbol{a}_{\mathcal{H}}$ and the learning of objective cost functions in deterministic measure via inverse reinforcement learning.

## 2.4 Interaction-Aware Probabilistic Prediction

When the deterministic measure is used, the robot's behavior generation policy in (2.14) reduces into an interaction-aware planning problem as in [119]. It enables the robot to leverage its influence to the human by considering the human's optimal response when it determines what actions to take.

When the EUT-based measure is utilized, (2.14) becomes

$$\begin{aligned}
\boldsymbol{a}_{\mathcal{R}}^{*} &= \arg\min_{\mathbf{a}_{\mathcal{R}}} C_{\mathcal{R}}\left(x^{t}, \mathbf{a}_{\mathcal{R}}, \pi_{\mathcal{H}}(x^{t}, \mathbf{a}_{\mathcal{R}}, \boldsymbol{x}_{env}^{t}; \theta_{\mathcal{H}}), \boldsymbol{x}_{env}^{t}; \theta_{\mathcal{R}}\right) \\
&= \arg\min_{\mathbf{a}_{\mathcal{R}}} \mathbb{E}_{\boldsymbol{x}\sim p_{x}, \boldsymbol{x}_{env}\sim p_{x_{env}}, \boldsymbol{a}_{\mathcal{R}}\sim p_{a_{\mathcal{R}}}} \left[\sum_{k=0}^{N-1} c_{\mathcal{H}}\left(x^{t,k}, a_{\mathcal{R}}^{k}, a_{\mathcal{H}}^{k}, x_{env}^{t,k}; \theta_{\mathcal{H}}\right)\right] \\
&= \arg\min_{\mathbf{a}_{\mathcal{R}}} \mathbb{E}_{\boldsymbol{x}\sim p_{x}, \boldsymbol{x}_{env}\sim p_{x_{env}}} \left[P\left(\mathbf{a}_{\mathcal{H}}, | x^{t}, \mathbf{a}_{\mathcal{R}}, \boldsymbol{x}_{env}^{t}, \theta_{\mathcal{H}}\right) \sum_{k=0}^{N-1} c_{\mathcal{H}}\left(x^{t,k}, a_{\mathcal{R}}^{k}, a_{\mathcal{H}}^{k}, x_{env}^{t,k}; \theta_{\mathcal{H}}\right)\right].
\end{aligned} \tag{2.26}$$

Compared to the deterministic measure, the EUT-based measure can describe more realistic interaction between the human and the robot. Instead of considering only the optimal responses from the human, the robot should consider the distribution over human's future actions given the robot's actions and environment states.

From the prediction perspective of view, $P\left(\mathbf{a}_{\mathcal{H}}, | x^t, \mathbf{a}_{\mathcal{R}}, \boldsymbol{x}^t_{env}, \theta_{\mathcal{H}}\right)$ in (2.26) has defined an *interaction-aware probabilistic prediction* problem. Compared to other prediction problems that try to predict future actions of agents based on only historical data (including both historical system states and environment states) [72, 144, 71, 121, 38] as shown in (2.27), the *interaction-aware probabilistic prediction* can directly model the mutual influence between two agents.

$$P\left(\mathbf{a}_{\mathcal{H}}, | x^t, \boldsymbol{x}^t_{env}, \theta_{\mathcal{H}}\right) \tag{2.27}$$

## 2.5 Inverse Reinforcement Learning

Initially proposed by Kalman [56], the concept of Inverse Reinforcement Learning (IRL) is first formulated in [92]. It aims to infer the reward/cost functions of agents from their observed behavior by assuming that the agents are rational. To deal with uncertainties and noisy observations, Ziebart et al. [166] extended the algorithm based on the principle of maximum entropy. It assumes that agents actions/behavior with lower cost are exponentially more probable, and thus an exponential distribution family can be established to approximate the distribution of actions/behavior. Building on this, Levine et al. [74] formulated the continuous IRL algorithm and used it to minic and predict human driving behavior. In this section, we review the continuous-domain IRL algorithm developed in [74].

For a system with state $x$ and action $a$, we assume that cost function is parameterized as a linear combination of features denoted by vector $\phi$:

$$c(x^t, a^t; \theta) = \theta^T \phi(x^t, a^t). \tag{2.28}$$

Then over a trajectory with length $N$, denoted as

$$\xi = [(x^0, a^0), (x^1, a^1), \cdots, (x^{N-1}, a^{N-1}), x^N] = (x^0, \boldsymbol{a}),$$

the cumulative cost function becomes

$$C(x^0, \boldsymbol{a}; \theta) \ = \ \theta^T \sum_{t=0}^{N-1} \phi(x^t, a^t) = \theta^T \boldsymbol{\Phi}(x^0, \boldsymbol{a}) \tag{2.29}$$

The goal of IRL is to find the weights $\theta$ which maximizes the likelihood of the demonstrations set $\mathcal{U}_D \triangleq \{\xi_i\}$ with $i$=1, $\cdots$, M:

$$\theta^* = \arg\max_{\theta} P(\mathcal{U}_D | \theta) \tag{2.30}$$

where M is the number of demonstrations in the set. Building on the principle of maximum entropy, we assume that trajectories are exponentially more likely when they have lower cost:

$$P(\boldsymbol{a},\theta) \quad \propto \quad \exp\left(-\beta C(x^0,\boldsymbol{a};\theta)\right), \tag{2.31}$$

$$P(\boldsymbol{a}|\theta) \quad = \quad \frac{\exp\left(-\beta C(x^0,\boldsymbol{a};\theta)\right)}{\int \exp\left(-\beta C(x^0,\boldsymbol{a};\theta)\right) d\boldsymbol{a}} \tag{2.32}$$

Thus the probability (likelihood) of the demonstration set becomes

$$P(\mathcal{U}_D|\theta) = \Pi_{i=1}^{M}\frac{P(\xi_i^D,\theta)}{P(\theta)} = \Pi_{i=1}^{M}\frac{P(\boldsymbol{a}_i^D,\theta)}{P(\theta)} = \Pi_{i=1}^{M}\frac{P(\boldsymbol{a}_i^D,\theta)}{\int P(\tilde{\boldsymbol{a}},\theta)d\tilde{\boldsymbol{a}}}. \tag{2.33}$$

To tackle the partition term $\int P(\tilde{\boldsymbol{a}},\theta)d\tilde{\boldsymbol{a}}$ in (2.33), we approximate $C(x^0,\boldsymbol{a}_{\mathcal{R}},\tilde{\boldsymbol{a}}_{\mathcal{H}};\theta)$ with its Laplace approximation as proposed in [73]:

$$C(x^0,\tilde{\boldsymbol{a}};\theta) \quad \approx \quad C(x^0,\boldsymbol{a}_i;\theta) + \left(\tilde{\boldsymbol{a}}-\boldsymbol{a}_i^D\right)^T \frac{\partial C}{\partial \boldsymbol{a}}|_{\boldsymbol{a}_i^D} + \frac{1}{2}\left(\tilde{\boldsymbol{a}}-\boldsymbol{a}_i^D\right)^T \frac{\partial^2 C}{\partial \boldsymbol{a}^2}|_{\boldsymbol{a}_i^D}\left(\tilde{\boldsymbol{a}}-\boldsymbol{a}_i^D\right). \tag{2.34}$$

With the assumption of locally optimal demonstrations, we have $\frac{\partial C}{\partial \boldsymbol{a}}|_{\boldsymbol{a}_i^D}=0$ in (2.34). This simplifies the partition term $\int P(\tilde{\boldsymbol{a}},\theta)d\tilde{\boldsymbol{a}}$ as a Gaussian Integral where a closed-form solution exists (see [73] for details). Substituting (2.33) and (2.34) into (2.30) yields the optimal parameter $\theta^*$ as the maximizer.

We use the IRL algorithm to learn the cost function with deterministic measure in human-robot interaction. Namely, we try to learn $C_{\mathcal{H}}\left(x^t,\mathbf{a}_{\mathcal{R}},\mathbf{a}_{\mathcal{H}},\boldsymbol{x}_{env}^t;\theta_{\mathcal{H}}\right)$ defined in (2.15). In this case, we will collect demonstrations with two agents interacting with each other, one acting as the robot and the other as the human. We assume that the human can accurately observe $\boldsymbol{a}_{\mathcal{R}}$ while the human generates $\boldsymbol{a}_{\mathcal{H}}$. Thus, the key equation in (2.32) is specified as

$$P(\boldsymbol{a}_{\mathcal{H}}|x^t,\boldsymbol{a}_{\mathcal{R}},\boldsymbol{x}_{env}^t,\theta_{\mathcal{H}}) = \frac{\exp\left(-\beta C_{\mathcal{H}}\left(x^t,\mathbf{a}_{\mathcal{R}},\mathbf{a}_{\mathcal{H}},\boldsymbol{x}_{env}^t;\theta_{\mathcal{H}}\right)\right)}{\int \exp\left(-\beta C_{\mathcal{H}}\left(x^t,\mathbf{a}_{\mathcal{R}},\mathbf{a}_{\mathcal{H}},\boldsymbol{x}_{env}^t;\theta_{\mathcal{H}}\right)\right) d\boldsymbol{a}_{\mathcal{H}}}, \tag{2.35}$$

and the following steps in (2.33) and (2.34) adapt accordingly.

## 2.6 Datasets for driving behavior

Throughout this dissertation, we have utilized two open datasets for human driving behavior: the Next Generation SIMulation (NGSIM) dataset [6] dataset and the INTERnational, Adversarial and Cooperative moTION (INTERACTION) dataset [159]. We briefly introduce the two datasets as follows.

**Figure 2.4:** A camera recording trajectories on highways in NGSIM. *Image source: https://www.fhwa.dot.gov/publications/research/operations/its/06135/index.cfm*

### 2.6.1 NGSIM dataset

The NGSIM dataset is a dataset gathered by the Federal Highway Administration (FHWA) since 1970s. It used digital video cameras mounted on top of buildings to overlook and record vehicle trajectories on highways and intersections, as shown in Fig. 2.4 [6]. It includes vehicle trajectories at three locations: the interstate 80 Freeway dataset, the Lankershim Boulevard dataset and the US Highway 101 dataset. In each trajectory, the $(x, y)$ coordinates, speed, and acceleration are provided at each time step with a sampling frequency of 10Hz. In this dissertation, for the work in Chapter 3 and Chapter 4, we use the interstate 80 Freeway dataset.

### 2.6.2 INTERACTION dataset

The INTERACTION dataset is a dataset constructed by us in 2019 with drone cameras and fixed cameras. It contains naturalistic motions of various traffic participants in a variety of highly interactive driving scenarios from four different countries including the United States, China, Germany and Bulgaria [159]. The driving scenarios include 11 locations with four geometric categories: roundabout, un-signalized intersection, signalized intersection, merging and lane changing. Different from the NGSIM dataset, the INTERACTION dataset focuses more on the interactive driving behavior of vehicles where many near-collision and negotiation scenarios are contained. For each trajectory, the $(x, y)$ coordinates, speed, acceleration and yaw angle are provided with a sampling frequency of 10Hz. Besides those, the INTERACTION dataset also provides a high-definition map for each location, defined in the format of lanelet2 [105]. In each map, we include three layers: the physical layer such as the lane markers and curbs, the semantic layer such as the traffic signs, and the planning-rated layer which formulates the drivable blocks in each lane and the logical relationships among them. Two exemplar scenarios are shown in Fig. 2.5. For more details, one can refer [159].

In this dissertation, we use the "USA_Roundabout_FT" scenario for the work in Chapter 5, Chapter 7 and Chapter 8.

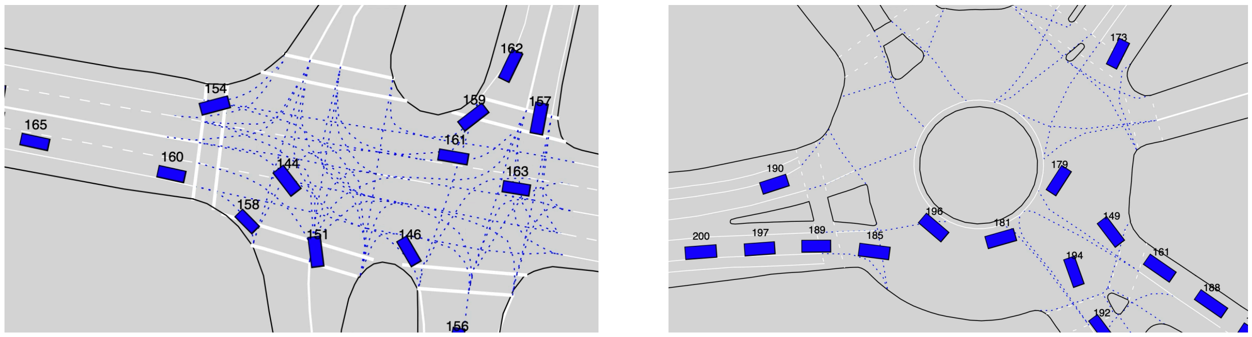


**Figure 2.5:** Two exemplar scenarios in the INTERACTION dataset: the left one is an intersection, and the right one is an roundabout. The blue rectangles represent vehicles, the white lines represent the physical layers on the road, the black lines are the curbs, and the dotted blue lines are the planning-based drivable lane separators defined via lanelet2.

## 2.7 Chapter Summary

In this chapter, we presented the game-theoretic framework and the simplified interaction-aware model utilized in this dissertation to formulate the human-robot interaction. We represented human behavior via cost functions under the deterministic measure, the EUT measure and the CPT measure to capture its diversity in terms of causality and irrationality. Moreover, we discussed the formulation of the interaction-aware probabilistic prediction, and the IRL algorithm to learn the cost functions from human demonstrations. Finally, we briefly introduce the two dataset utilized throughout the dissertation.

Since we have established the cost functions as effective representation for human behavior, in the next three chapters, i.e., Chapters 3-5, we will focus on the design of reward functions to describe different aspects of human behavior such as social compliance, hierarchy and irrationality. The corresponding learning algorithm/process for each of them will also be introduced.

# Chapter 3

# Courtesy Behavior Learning and Generation

To enable autonomous systems to interact with human in a human-like manner, we should design socially-compliant reward functions for them to optimize. In this chapter, we will address this problem, using autonomous vehicles as an application example.

## 3.1 Introduction

As discussed in Chapter 2, we generate the behavior of autonomous systems by optimizing designed cost functions in the MPC framework. Traditionally, when designing the robot's cost function, we focus on the terms of the robot's interest. For instance, for autonomous cars, we care about the safety, efficiency and driving quality if a passenger is on board. Arguably, such consideration alone is rather *selfish*.

Selfishness is not a problem for systems that work isolated without interacting with others. Also, it may not be a problem for autonomous systems that behave conservatively. Such conservative systems let the human move first during the interaction. They predict all possible human trajectories assuming themselves as not being there, and react to the worst scenario. Hence, such conservativeness makes robots always try to stay out of the way and let people do first what they want. As we improve the efficiency of autonomous systems, however, by adopting game theoretic approaches, the autonomous systems may take advantage of its influence on humans. Under such circumstances, if the reward function is not carefully designed, a selfish robot will make more aggressive movements around human, as illustrated in Fig. 3.1. For instance, a robot car can cut people off, or inch forward at intersections to go first [118][119].

Hence, as we are getting better at solving the optimization problem by utilizing better models of the world and of the people in it, there is an increased requirement for the cost function we optimize to capture what we want. We propose that purely selfish robots that care about their safety and driving quality are not good enough. They should also

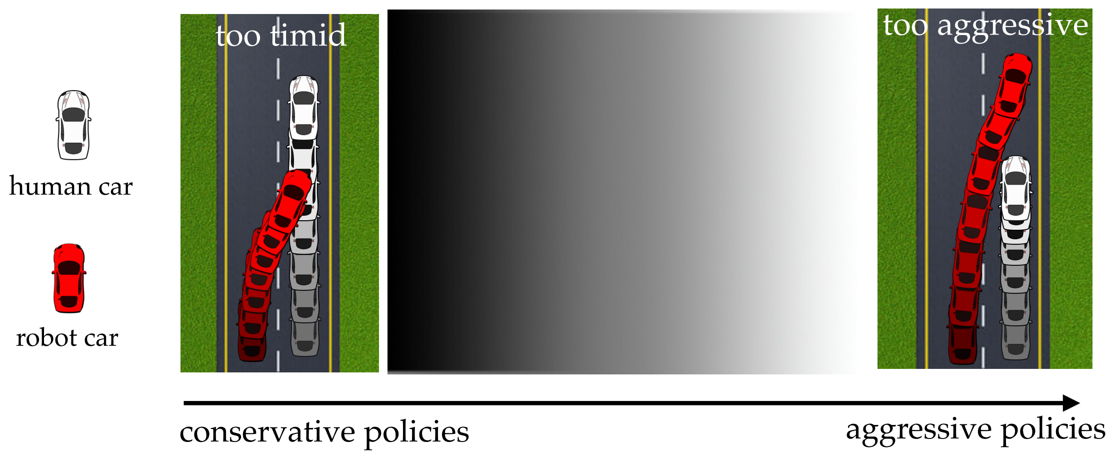


**Figure 3.1:** As we get better at optimizing the efficiency of the robots by actively leveraging their influence to other human, a selfish reward design might generate too aggressive behavior which makes the interaction with human not safe and not socially compliant.

be *courteous* to others. This is of crucial importance since humans are not perfectly rational, and one of the irrationality is that they view utilities in a relative format with a reference point instead of an absolute format, and they weigh loss higher than gains when evaluating actions[146]. Hence, if a robot's action brings too much loss to the human's utilities compared to the alternatives (the reference point) in his/her mind, human tend to choose more defensive and non-cooperative actions to minimize his/her loss.

*We advocate that a robot should balance minimizing the inconvenience it brings to other people, and that we can formalize inconvenience as the increase in the other people's cost due to the robot's behavior.*

In this chapter, our contributions are as follows:

**A formalism for courtesy incorporating irrational human behavior.** We formalize courteous planning as a trade off between the robot's selfish objective and a courtesy term, and introduce a mathematical definition for this term for irrational human behavior – we measure the increase of the vehicle's best cost under the robot's planned behavior, compared to the vehicle's best cost under an alternative "best case scenario", and define the cost increase as the courtesy term.

**An analysis of the effects of courteous planning.** We show the difference between courteous and selfish robots under different traffic scenarios. The courteous robot leaves the person more space when it merges, and might even block another agent (not a person) to ensure that the human can safely proceed.

**Showing that courtesy helps explain the social compliance in human driving.** We do an Inverse Reinforcement Learning (IRL)-based analysis [73, 1, 166, 2] to study whether our courtesy term helps in better predicting how humans drive. On the NGSIM dataset [6] of real human driver trajectories, we find that courtesy produces trajectories that are significantly closer to the ground truth.

## 3.2 A Two-Player Game

Following the game-theoretic framework in Chapter 2, we formulate the human-robot interaction as a two-player game in this chapter, with $\mathcal{R}$ as the robot and $\mathcal{H}$ as a human.

Let $x_{\mathcal{R}}$ and $u_{\mathcal{R}}$ denote, respectively, the robot's state and control input, and $x_{\mathcal{H}}$ and $u_{\mathcal{H}}$ for the human's. $x{=}(x_{\mathcal{R}}^T, x_{\mathcal{H}}^T)^T$ represents the states of the interaction system. For each agent, we have

$$x_{\mathcal{R}}^{t+1} = f_{\mathcal{R}}\left(x_{\mathcal{R}}^t, u_{\mathcal{R}}^t\right), \tag{3.1}$$

$$x_{\mathcal{H}}^{t+1} = f_{\mathcal{H}}\left(x_{\mathcal{H}}^t, u_{\mathcal{H}}^t\right), \tag{3.2}$$

and the overall system dynamics are

$$x^{t+1} = f\left(x^t, u_{\mathcal{R}}^t, u_{\mathcal{H}}^t\right). \tag{3.3}$$

We assume that both the human driver and the autonomous car are optimal planners, and they use Model Predictive Control (MPC) with a horizon of length $N$. Let $C_{\mathcal{R}}$ and $C_{\mathcal{H}}$ be, respectively, the cost functions of the robot car and the human driver over the horizon:

$$C_i\left(x^t, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}}; \theta_i\right) = \sum_{k=0}^{N-1} c_i\left(x^{t,k}, u_{\mathcal{R}}^k, u_{\mathcal{H}}^k; \theta_i\right), i{\in}\{\mathcal{R}, \mathcal{H}\} \tag{3.4}$$

where $\mathbf{u}_i{=}(u_i^0, u_i^1, \cdots, u_i^{N-1})^T$ are sequences of control actions of the robot car ($i{=}\mathcal{R}$) and the human driver ($i{=}\mathcal{H}$), and $x^{t,k}$ with $k{=}0, 1, \cdots, N{-}1$ are the corresponding sequence of system states. $\theta_i$ represent, respectively, the preferences of the robot car ($i{=}\mathcal{R}$) and the human driver ($i{=}\mathcal{H}$). At every time step $t$, the robot car and the human driver generate their optimal sequences of actions $\mathbf{u}_{\mathcal{R}}^*$ and $\mathbf{u}_{\mathcal{H}}^*$ by minimizing $C_{\mathcal{R}}$ and $C_{\mathcal{H}}$, respectively, execute the first steps $u_{\mathcal{R}}^{*0}$ and $u_{\mathcal{H}}^{*0}$ (i.e., set $u_i^t{=}u_i^{*0}$ in (3.3)), and replan for step $t{+}1$.

Such an optimization-based state feedback strategy formulates the closed-loop dynamics of the robot-human interaction system as a game. To simplify the game, we assume that the robot car has access to $C_{\mathcal{H}}$, and that the human only computes a best response to the robot's actions rather than trying to influence them, as in [119]. This means that the robot car can compute, for any control sequence it considers, how the human would respond and what cost the human will incur:

$$\begin{aligned} \mathbf{u}_{\mathcal{H}}^* &= \arg\min_{\mathbf{u}_{\mathcal{H}}} C_{\mathcal{H}}\left(x^t, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}}; \theta_{\mathcal{H}}\right) \triangleq g(x^t, \mathbf{u}_{\mathcal{R}}; \theta_{\mathcal{H}}) & (3.5) \\ C_{\mathcal{H}}^*(\mathbf{u}_{\mathcal{R}}) &= C_{\mathcal{H}}\left(x^t, \mathbf{u}_{\mathcal{R}}, \pi_{\mathcal{H}}(x^t, \mathbf{u}_{\mathcal{R}}; \theta_{\mathcal{H}}); \theta_{\mathcal{H}}\right). & (3.6) \end{aligned}$$

Here $\pi_{\mathcal{H}}(x^t, \mathbf{u}_{\mathcal{R}}; \theta_{\mathcal{H}})$ represents the response curve of the human driver towards the autonomous car.

Armed with this model, the robot can now compute what it should do, such that when the human responds, the combination is good for the robot's cost:

$$\mathbf{u}_{\mathcal{R}}^{*} = \arg\min_{\mathbf{u}_{\mathcal{R}}} C_{\mathcal{R}}\left(x^{t}, \mathbf{u}_{\mathcal{R}}, \pi_{\mathcal{H}}(x^{t}, \mathbf{u}_{\mathcal{R}}; \theta_{\mathcal{H}}); \theta_{\mathcal{R}}\right). \tag{3.7}$$

Hence, the next question is how to design an appropriate cost function such that courteous robot behavior can be generated via (3.6). Our observation is that a courteous robot car should care about not only its own interest such as speed and comfort, but also the potential inconvenience it brings to the human driver's utilities, and generates trajectories that are socially predictable and acceptable. In the following sections, we do so by integrating such inconvenience into the cost function of the robot.

## 3.3 Courteous Planning

We propose a *courteous planning* strategy that a courteous robot car should balance the minimization of its own cost function and the inconvenience (loss) it brings to the human driver. Therefore, we construct $C_{\mathcal{R}}$ in (3.7) as

$$C_{\mathcal{R}}\left(x^{t},\mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}};\theta_{\mathcal{R}},\theta_{\mathcal{H}},\lambda_c\right) = C_{\mathcal{R}}^{\text{self}}\left(x^{t},\mathbf{u}_{\mathcal{R}},\mathbf{u}_{\mathcal{H}};\theta_{\mathcal{R}}\right) + \lambda_c C_{\mathcal{R}}^{\text{court}}\left(x^{t}, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}}; \theta_{\mathcal{H}}\right), \tag{3.8}$$

where $C_{\mathcal{R}}^{\text{self}}$ is the cost function for a regular (selfish) robot car which cares about only its own utilities (safety, efficiency, etc), and $C_{\mathcal{R}}^{\text{court}}$ models the courtesy term of the robot car to the human driver. It is a function of the robot car's behavior, the human's behavior, the human's cost parameters ($\theta_{\mathcal{H}}$) and some alternative costs (see Section 3.3.A). $\lambda_c \in [0, \infty)$ captures the trade-off. If we want the robot car to be just as courteous as a human driver, we could learn $\lambda_c$ from human driver demonstration, as we do in Section 3.5. As robot designers, we might set this parameter higher than regular human driving to enable more courteous autonomous cars, particularly when they do not have passengers on board.

### 3.3.1 Alternative Costs

With any robot plan $\mathbf{u}_{\mathcal{R}}$, the robot car changes the human driver's environment and therefore induces a best cost for the human, $C_{\mathcal{H}}^{*}(\mathbf{u}_{\mathcal{R}})$. Our courtesy term compares this cost with the *alternative*, $C_{\mathcal{H}}^{\text{alt},*}$ – the best case scenario for the person. It is not immediately clear how to define this best case scenario since it may vary depending different on driving scenarios. We explore three alternatives.

**What the human could have done, had the robot car not been there.** We first consider a world in which the robot car wouldn't even exist to interfere the person. In such a world, the person gets to optimize their cost without the robot car:

$$C_{\mathcal{H}}^{\text{alt},*}(x^{t}, \theta_{\mathcal{H}}) = \min_{\mathbf{u}_{\mathcal{H}}} C_{\mathcal{H}}(x^{t}, \mathbf{u}_{\mathcal{H}}; \theta_{\mathcal{H}}) \tag{3.9}$$

This induces a very generous definition of courtesy: the alternative is for the robot car to not have been on the road at all. In reality though, the robot car is there, which leads to our second alternative.

**What the human could have done, had the robot car only been there to help the human.** Our second alternative is to assume that the robot car already on the road could be completely altruistic. The robot car could actually optimize the human driver's cost, being a perfect collaborator:

$$C_{\mathcal{H}}^{alt,*}(x^t, \theta_{\mathcal{H}}) = \min_{\mathbf{u}_{\mathcal{H}}, \mathbf{u}_R} C_{\mathcal{H}}(x^t, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}}; \theta_{\mathcal{H}}) \tag{3.10}$$

For this alternative, the robot car and the human would perform a joint optimization for the human's cost. For example, the robot car can brake to make sure that the human could change lanes in front of it, or even block another traffic participant to make sure the human has space.

**What the human could have done, had the robot car just kept doing what it was previously doing.** A fully collaborative robot car is still perhaps not the fairest one to compute inconvenience against. After all, the autonomous car does have a passenger sometimes, and it is fair to take their needs into account too. Our third alternative computes how well the human driver could have done, had the robot car kept acting the same way as it was previously doing:

$$C_{\mathcal{H}}^{alt,*}(x^t, \theta_{\mathcal{H}}) = \min_{\mathbf{u}_{\mathcal{H}}} C_{\mathcal{H}}(x^t, \mathbf{u}_{\mathcal{R}}^{t-1}, \mathbf{u}_{\mathcal{H}}; \theta_{\mathcal{H}}) \tag{3.11}$$

This means that the person is now responding to a constant robot trajectory $\mathbf{u}_{\mathcal{R}}^{t-1} = (u_{\mathcal{R}}^{t-1}, .., u_{\mathcal{R}}^{t-1})$, for instance, maintaining its current velocity.

Our experiments below explore these three different alternative options for the courtesy term.

### 3.3.2 Courtesy Term

We define the courtesy term based on the difference between what cost the human has, and what cost they would have had in the alternative, as illustrated in Fig. 3.2.

**Definition 1** (Courtesy of the Robot)

$$C_{\mathcal{R}}^{court}(x^t, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}}; \theta_{\mathcal{H}}) = \max\{0, C_{\mathcal{H}}(x^t, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}}; \theta_{\mathcal{H}}) - C_{\mathcal{H}}^{alt,*}(x^t; \theta_{\mathcal{H}})\} \tag{3.12}$$

Note that we could have also sent the courtesy term to simply be the human cost, and have the robot trade off between its cost and the human's. However, that would have penalized the robot for any cost the human incurs, even if the robot does not bring any inconvenience to the human. That might cause too conservative behavior. In fact, if we treat the alternative cost as the reference point in Prospect Theory – a human irrationality model [146], then the theory suggests that human weigh losses more than

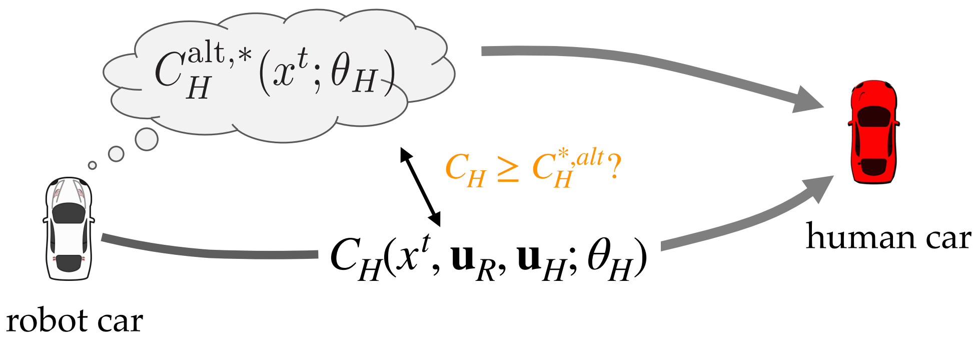

$C_H^{\text{alt},*}(x^t; \theta_H)$
$C_H \geq C_H^{*,alt}$?
human car
$C_H(x^t, \mathbf{u}_R, \mathbf{u}_H; \theta_H)$
robot car

**Figure 3.2:** An illustration of the courtesy term defined in (3.12). The robot car will compare two of the human's utilities: the alternative utilities and the responsive utilities caused by responding to the robot's actions. If the responsive utility is larger than the alternative utility, the action of the robot is bringing additional cost to the human and the robot should be aware of that and balance it in its optimization

gains. This means that our courteous robot car should care more about avoiding additional inconvenience, rather than providing more convenience, i.e., helping to reduce the human cost lower than the alternative one. Mathematically, this concept is formulated via Definition 1: the robot does not get any bonus for bringing the human cost lower than $C_{\mathcal{H}}^{\text{alt},*}$ (possible with some definitions of $C_{\mathcal{H}}^{\text{alt},*}$), it only gets a penalty for making it higher.

### 3.3.3 Solution

Thus far, we have constructed a compound cost function $C_{\mathcal{R}}(x^t, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}}; \theta_{\mathcal{R}}, \theta_{\mathcal{H}}, \lambda_c)$ to enable a courteous robot car, considering three alternative costs. At every step, the robot needs to solve the optimization problem in (3.7) to find the best actions to take. We approximate the solution by alternatively fixing one of $\mathbf{u}_{\mathcal{R}}$ or $\mathbf{u}_{\mathcal{H}}$, and solving for the other. This problem can also be solved as a nested optimization problem as in [119], and you can find more details regarding the solutions there.

## 3.4 Analysis of Courteous Planning

In this section, we analyze the effect of courteous planning on the robot's behavior in different simulated driving scenarios. In Section 3.5, we study how courteous planning can help better explain real human driving data, enabling robots to be more human-like and predictable, as well as better able at anticipating human driver actions on the road. *Simulation Environment*: We implement the simulation environment using Julia [46] on a 2.5 GHz Intel Core i7 processor with 16 GB RAM. We set the horizon length to N=10, and the sampling time to 0.1s. Our simulated environment is 1/10 scale of the real world: 1/10 road width, car sizes, maximum acceleration (0.5m/s$^2$) and deceleration (-1.0m/s$^2$), and low speed limit (1.0m/s).

Regarding the cost functions $C_{\mathcal{H}}$ and $C_{\mathcal{R}}$ in (3.6)-(3.8), except for the courtesy term formulated above, we penalize safety, car speed, comfort level and goal distances in both $C_{\mathcal{H}}$ and $C_{\mathcal{R}}^{self}$. Details about this can be found later in Section 3.5.

For all results, we denote a selfish (baseline) autonomous car with gray rectangle, a courteous one as orange, and the human driver as dark blue.

### 3.4.1 The Effect of Courtesy

#### Lane Changing

We first consider a lane changing driving scenario, as shown in Fig. 3.3 and Fig. 3.4. The autonomous car wants to merge into the human driver's lane from an adjacent lane. We assume that the goal of the human driver is to maintain speed. Then all three different alternatives lead to the same alternative optimal behavior and cost of the human: the human would go in their lane undisturbed by the robot. Hence, with constant $C_{\mathcal{H}}^{alt,*}$, we focus on the influence of the trade-of factor $\lambda_c$ in the results.

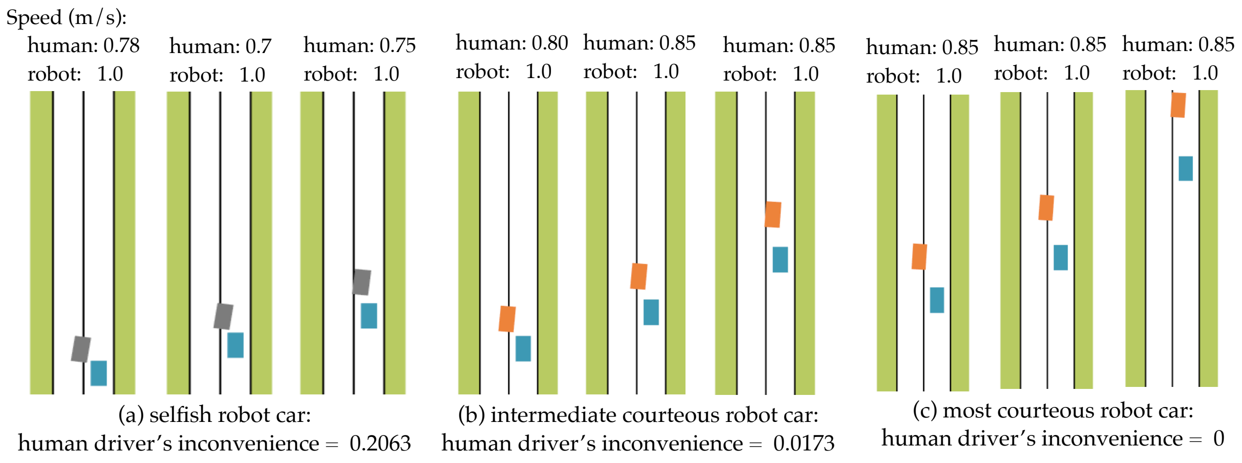


**Figure 3.3:** A lane changing scenario: both the human car and robot car speed at 0.85 m/s initially; (a) a selfish robot car (grey) merges in front of the human (blue) with a small gap so that the human brakes to yield; (b) an intermediate courteous robot car (orange) merges with a larger gap, which releases the human driver from hard brakes; (c) a most courteous robot car (orange) merges with a gap large enough so that the human can maintain speed.

We present two sets of simulation results in Figs. 3.3 - 3.5, where the initial human driver's speeds are 0.85 m/s and 0.9 m/s respectively. The results show that as $\lambda_c$ increases, i.e., being more courteous, the autonomous car tends to leave a larger gap when it merges in front of the human, and the human brakes less (Fig. 3.3 from left to right). When the human driver's initial speed is high enough, a courteous autonomous car decides to merge afterwards instead of cutting in, as shown in Fig. 3.5.

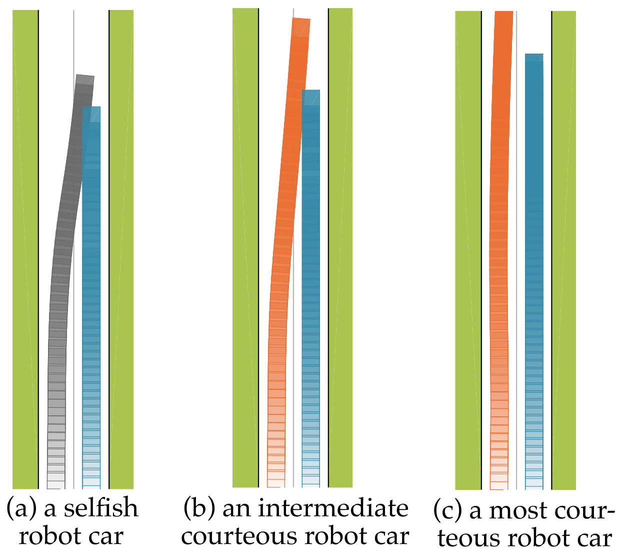

(a) a selfish robot car
(b) an intermediate courteous robot car
(c) a most courteous robot car

**Figure 3.4:** The overview of the trajectories for the lane changing scenario in Fig. 3.3. A most courteous robot car (orange) merges with the largest gap, minimizing possible inconvenience brought to the human.

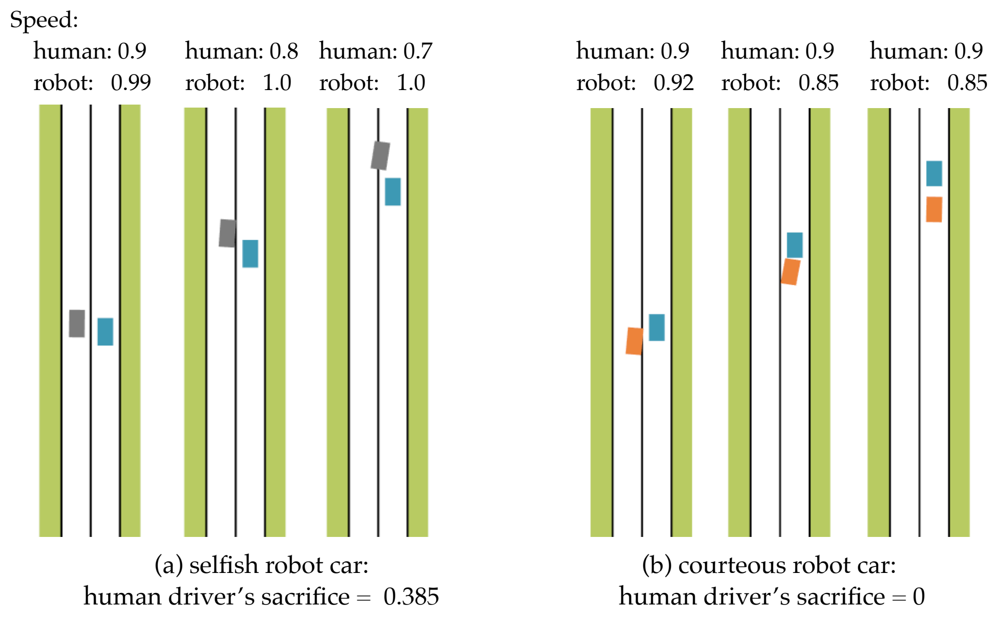


**Figure 3.5:** Another lane changing scenario: both the human car and robot car speed at 0.9 m/s initially; (a) a selfish robot car (grey) accelerates and merges in front of the human driver (blue) with a small gap, scaring the human driver to brake; (b) a courteous robot car (orange) decelerates and merges after the human driver so that the human can maintain speed.

Figure 3.6 summarizes the relationship between the human driver's inconvenience (the magnitude of the courtesy term) and $\lambda_c$ for the simulation conditions in Fig. 3.3.

One can note that as the courtesy of the autonomous car increases, the human driver's inconvenience decreases.

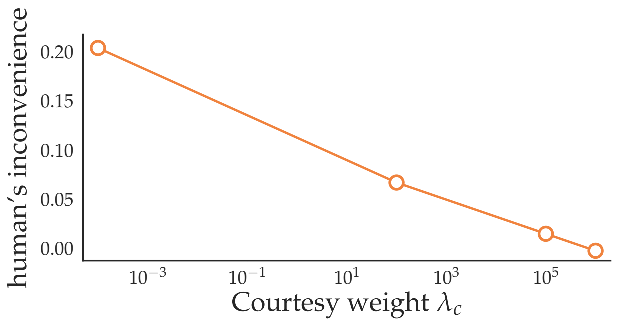
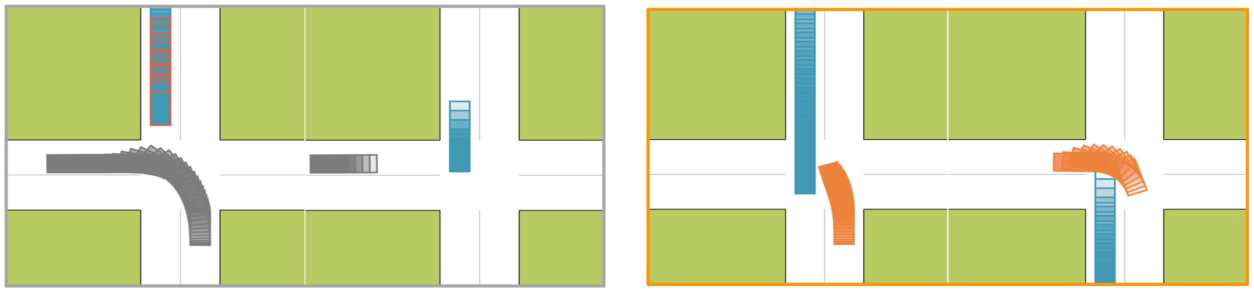


**Figure 3.6:** Inconvenience to the human decreases as the robot increases the courtesy weight $\lambda_c$ in its cost function.

## Turning Left

In this scenario, an autonomous car wants to take a left turn at an intersection with a straight-driving human. In this case as well, the alternative behaviors that we consider when evaluating inconvenience are the same among three different alternatives: the human driver crosses the intersection maintaining speed.

Simulation results with a courteous and selfish autonomous car are shown in Fig. 3.7, where a selfish robot car takes a left turn immediately and forces the human driver to brake (Fig. 3.7(a)); while a courteous robot car waits in the middle of the intersection and takes the left turn after the human driver passes the intersection so that the human can maintain its speed (Fig. 3.7(b)).



(a) a selfish robot car takes the left turn first and forces the human driver to brake

(b) a courteous robot car waits until the human driver passes the intersection

**Figure 3.7:** Interaction between a straight-driving human and a left-turning autonomous car: (a) a selfish (baseline) robot car takes a left turn immediately and forces the human driver to brake (red frames); (b) a courteous robot car waits in the middle of the intersection and takes the left turn after the human passes so that the human can maintain speed.

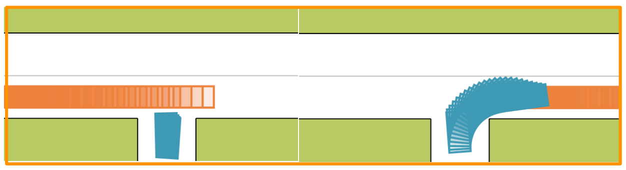
### 3.4.2 Influence of Different Alternative Costs for Evaluating Inconvenience

In the previous examples, the human would have arrived at the same trajectory regardless of which alternative world we are considering to evaluate how much inconvenience the autonomous car is causing. Here, we consider a scenario in which that is no longer the case to highlight the differences generated by the alternative formulations of courtesy in the robot car's behavior.

We consider a scenario where the human is turning right, with a straight-driving robot car coming from their left. In this scenario, the three alternative costs are different, which leads to different courtesy terms:

- Alternative I–Robot car not being there: the optimal human behavior would be to take a right turning directly;
- Alternative II–Robot car being collaborative: the robot would take the necessary yielding maneuver to let the human driver take the right turn first, leading to the same alternative optimal human behavior of performing the right turn directly;
- Alternative III–Robot car maintaining behavior: the robot car would maintain its speed, and the optimal human behavior would be to slow down.

Figure 3.8 summarizes the results of using these different courtesy terms. In Alternative III, a courteous robot car goes first, as shown in Fig. 3.8(a). Intuitively, this is because $C_{\mathcal{H}}^{\mathrm{alt},*}$ is initially high, and by maintaining its speed (or even accelerating depending on $C_{\mathcal{R}}^{\mathrm{self}}$), no further inconvenience is brought to the human by the robot car, i.e., $C_{\mathcal{R}}^{\mathrm{court}}$ remains zero. Hence, the robot car goes first (Had the robot try to brake, it only increases $C_{\mathcal{R}}^{\mathrm{self}}$ without changing $C_{\mathcal{R}}^{\mathrm{court}}$=0, and therefore $C_{\mathcal{R}}$ increases). The other two alternatives (I and II) are much more generous to the human. Results in Fig. 3.8(b) show that a courteous robot car finds it too expensive to force the human to go second, and slows down to let the human go first. The red frames in Fig. 3.8(b) indicate the time instants when the autonomous car brakes.


(a) a courteous robot car forces the human to go second

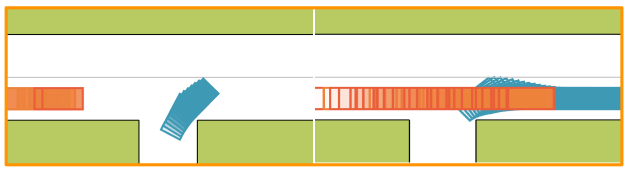
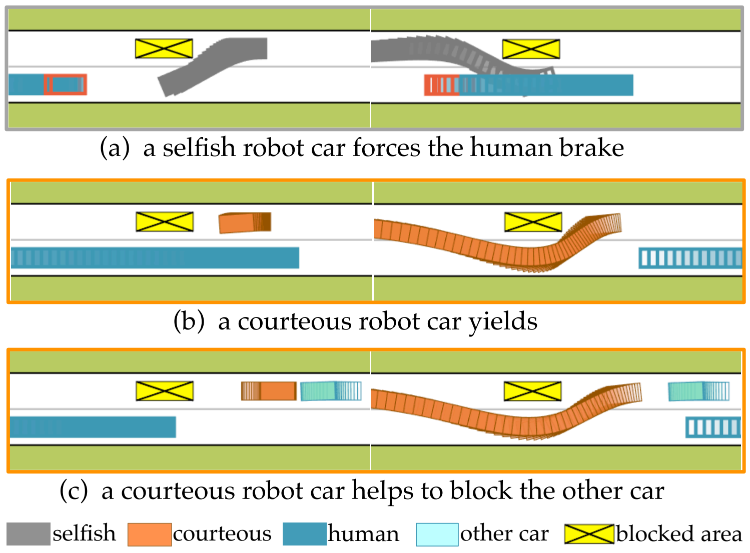

(b) a courteous robot car yields and let the human go first

**Figure 3.8:** Interaction between a right-turning human driver and a courteous autonomous car with different alternatives when evaluating the courtesy terms: (a) the robot car goes first when it evaluates the courtesy term using going forward as an alternative world; (b) the robot car yields and let the human go first when it evaluates the courtesy term based on a collaborative or not-being-there alternative world.

### 3.4.3 Extension to environments with multiple agents

We study a scenario on a two-way road. The robot car and the human are driving towards opposite directions, but the robot car is blocked and it has to temporarily merge into the human driver's lane to get through, as in Fig. 3.9. We use the collaborative robot as our alternative formulation of the courtesy term in this scenario.



(a) a selfish robot car forces the human brake

(b) a courteous robot car yields

(c) a courteous robot car helps to block the other car

**Figure 3.9:** A blocking-area overtaking scenario: (a) with a selfish cost function, the robot car overtakes first and forces the human driver to brake; (b)(c) a courtesy-aware robot car yields to the human driver and even helps to block other cars depending on its formulation of the human driver's alternative world

When there are only two agents in the environment, i.e., the autonomous car and the human driver, the results for a selfish and a courteous autonomous car are shown in Fig. 3.9(a)-(b): A selfish autonomous car directly merges into the human's lane and forces the human driver to brake; while a courteous autonomous car decides to wait until the human driver passes by since the courtesy term becomes too expensive to go first.

Such courtesy-aware planning becomes much more interesting when there is a third agent in the environment, as shown in Fig. 3.9(c). We assume that the third agent is a responsive agent to the autonomous car and the autonomous car is courteous only to the human driver (and not to both). In this case, for $C_{\mathcal{H}}^{\mathrm{alt},*}$, the human would ideally want to pass undisturbed by either the robot or the other agent: the courtesy term captures the difference in cost to the human between the robot's behavior and the alternative of a collaborative robot, and this cost to the human depends on how much progress the human is able to make and how fast. As a result, a very courteous robot has an incentive to produce behavior that is as close as possible to making that happen.

Then an interesting behavior emerges: the autonomous car first backs up to block the third agent (the following car) from interrupting the human driver until the human driver safely passes them, and then the robot car finishes its task. This displays truly collaborative behavior, and only happens with high enough weight on the courtesy term. This may not be practical for real on-road driving, but it enables the design of highly courteous robots in some particular scenarios where human have higher priority over all other autonomous agents.

## 3.5 Courtesy Helps Explain Human Driving

Thus far, we have shown that courtesy is useful for enabling cars to generate actions that do not cause inconvenience to other drivers. We have also seen that the larger the weight we put on the courtesy term, the more the car behavior becomes social. A natural next question is – are humans courteous?

Our hypothesis is that our courtesy term can help explain human driving behavior. If that is the case, this has two important implications: it means that it can enable robots to better predict human actions by giving them a more accurate model of how people drive, and it also means that robot can use courtesy to produce more human-like driving.

We put our hypothesis to the test by learning a cost function from human driver data, with and without a courtesy feature. We find that using the courtesy feature leads to a more accurate cost function that is better at reproducing human driver data, lending support to our hypothesis.

### 3.5.1 Learning Cost Functions from Human Demonstrations

#### Human Data Collection

The human data is collected from the Next Generation SIMulation (NGSIM) dataset [6], which captures the highway driving behaviors/trajectories by digital video cameras mounted on top of surrounding buildings. We selected 153 left-lane-changing driving trajectories on Interstate 80 (near Emeryville, California), and separated them into two sets: a *training* set of size 100 (denoted by $\mathcal{U}_D$, i.e., the human demonstrations), and the other 53 trajectories as the *test* set.

#### Learning Algorithm

We use Inverse Reinforcement Learning (IRL) [1, 166, 2, 73] to learn an appropriate cost function from human data.

We assume that cost function is parameterized as a linear combination of features:

$$c(x^t, u^t_{\mathcal{R}}, u^t_{\mathcal{H}}; \theta) = \theta^\top \phi(x^t, u^t_{\mathcal{R}}, u^t_{\mathcal{H}}). \tag{3.13}$$

Then over the trajectory length L, the cumulative cost function becomes

$$C(x^0, \boldsymbol{u}_{\mathcal{R}}, \boldsymbol{u}_{\mathcal{H}}; \theta) = \theta^T \sum_{t=0}^{L-1} \phi(x^t, u^t_{\mathcal{R}}, u^t_{\mathcal{H}}) \triangleq \theta^T \Phi(x^0, \boldsymbol{u}_{\mathcal{R}}, \boldsymbol{u}_{\mathcal{H}}) \tag{3.14}$$

where $\boldsymbol{u}_{\mathcal{R}}$ and $\boldsymbol{u}_{\mathcal{H}}$ are, respectively, the actions of the robot car and the human over the trajectory. Our goal is to find the weights $\theta$ which maximizes the likelihood of the demonstrations:

$$\theta^* = \arg\max_{\theta} P(\mathcal{U}_D|\theta) \tag{3.15}$$

Building on the principle of maximum entropy, we assume that trajectories are exponentially more likely when they have lower cost:

$$P(\mathbf{u}_{\mathcal{H}}, \theta) \propto \exp\left(-C(x^0, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}}; \theta)\right). \tag{3.16}$$

Thus the probability (likelihood) of the demonstration set becomes

$$P(\mathcal{U}_D|\theta) = \Pi_{i=1}^{n} \frac{P(\mathbf{u}^D_{\mathcal{H},i}, \theta)}{P(\theta)} = \Pi_{i=1}^{n} \frac{P(\mathbf{u}^D_{\mathcal{H},i}, \theta)}{\int P(\tilde{\mathbf{u}}_{\mathcal{H}}, \theta) d\tilde{\mathbf{u}}_{\mathcal{H}}} \tag{3.17}$$

where $n$ is the number of trajectories in $\mathcal{U}_D$. For more details about the algorithm, one can refer to Section 2.5 in Chapter 2.

### 3.5.2 Experiment Design

**Hypothesis.** Within human interactions, human drivers show courtesy to others, i.e., they optimize a compound cost function in the form of $C = C^{self} + \lambda_c C^{court}$ as (3.8) instead of a selfish one as $C^{self}$.

**Independent Variable.** To test our hypothesis, we run two sets of IRL on the same set of human data, but with one different feature. For the selfish cost function $C^{self}$, four features are selected as follows:

- speed feature $f_d$: deviation of autonomous car's speed compared to the speed limit:

$$f_d = (v - v_d)^2 \tag{3.18}$$

- comfort features $f_{acc}$ and $f_{steer}$: jerk and steering rate of the autonomous car;
- goal feature $f_g$: distance to the target lane:

$$f_g = e^{\frac{d_g}{w_l}}, \tag{3.19}$$

  where $d_g$ is the Euclidean distance and $w_l$ is the lane width.

- safety feature $f_s$: relative positions with respect to surrounding cars;

$$f_s = \sum_{i=1}^{n_s} e^{-d_i}, \tag{3.20}$$

where $n_s$ is the number of surrounding cars and $d_i, i{=}0, 1, \cdots, n_s$ is the distance to each of them.

For the courtesy-aware cost function $C = C^{\text{self}} + \lambda_c C^{\text{court}}$, we use the same four features as above, plus one additional feature that equals to the courtesy term.
**Dependent Measures.** We measured the similarity between trajectories planned with the learned cost functions and human driving trajectories on the *test* set (another 53 left-lane changing scenarios that are different from the training set from the NGSIM dataset).

### 3.5.3 Analysis

**Training performance.** The training results are shown in Fig. 3.10 and Table 3.1. One can see that with the additional courtesy term, better learning performance (in terms of training loss) has been achieved. This is a sanity check: having access to one extra DOF can lead to better training loss regardless, but if it did not that would invalidate our hypothesis.

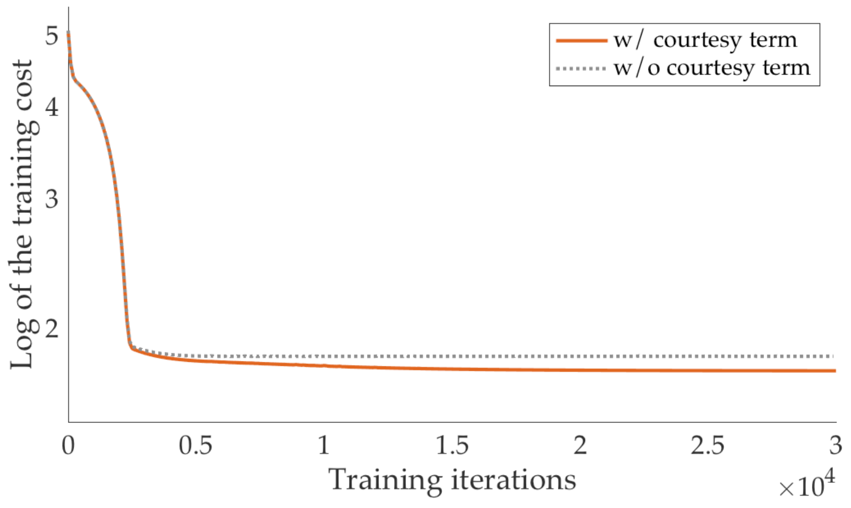


**Figure 3.10:** Training curves for cost functions with and without the courtesy term: with the courtesy term, the learning loss is much lower, meaning that adding the courtesy term as a feature can better explain the human demonstrations.

**Trajectory similarity.** Figure 3.11 shows one demonstrative example of the trajectories for a selfish car (grey) and a courteous car (orange), with four surrounding vehicles. The dark blue rectangle is the human driver in our two-agent robot-human interaction

| | $\theta_g$ | $\theta_d$ | $\theta_{acc}$ | $\theta_{steer}$ | $\theta_s$ | $\lambda_c$ |
|---|---|---|---|---|---|---|
| $C^{self}$ | 1.0 | 2.08e+04 | 5.80e+02 | 3.91e+02 | 4.37 | – |
| C | 1.0 | 1.96e+02 | 6.7e+04 | 2.36e+02 | 6.53 | 9.89e+04 |

**Table 3.1:** The parameters in C learned via IRL

system and all other vehicles (cyan) are treated as moving obstacles. It shows that a simulated car with C that includes courtesy manages to reduce its influence on the human driver by choosing a much smoother and less aggressive merging curve, while a car driven by $C^{self}$ merges in much aggressively.

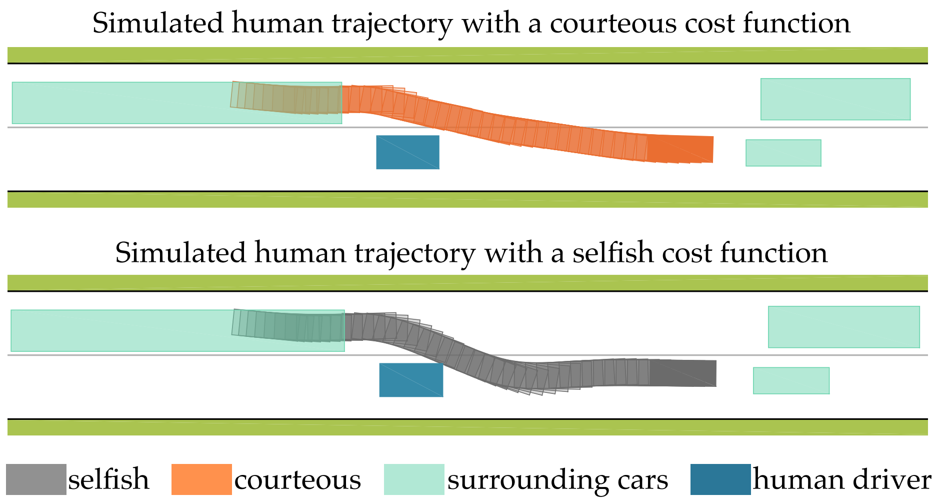

Simulated human trajectory with a courteous cost function
Simulated human trajectory with a selfish cost function
selfish
courteous
surrounding cars
human driver

**Figure 3.11:** An example pair of simulated trajectories with courteous (top) and selfish (bottom) cost functions. Compared to the selfish robot car, a courteous robot car can generate less aggressive manuveurs for better interaction with human.

Results for all 53 left-lane changing test trajectories are given in Fig. 3.12 (left). To describe the similarities among trajectories, we adopted the Mean Euclidean Distance (MED) [110]. As shown in Fig. 3.12 (right), the courtesy-aware trajectories are much similar to the ground truth trajectories, i.e., a courteous robot car behaves more human-like. We have also calculated the space headways of the following human driver on the robot car's target lane for all 53 test scenarios, and the statistical results are given in Fig. 3.12 (middle). Compared to a selfish robot car, a courteous robot car can achieve safer left-lane changing behaviours in terms of following gaps for the human driver behind.

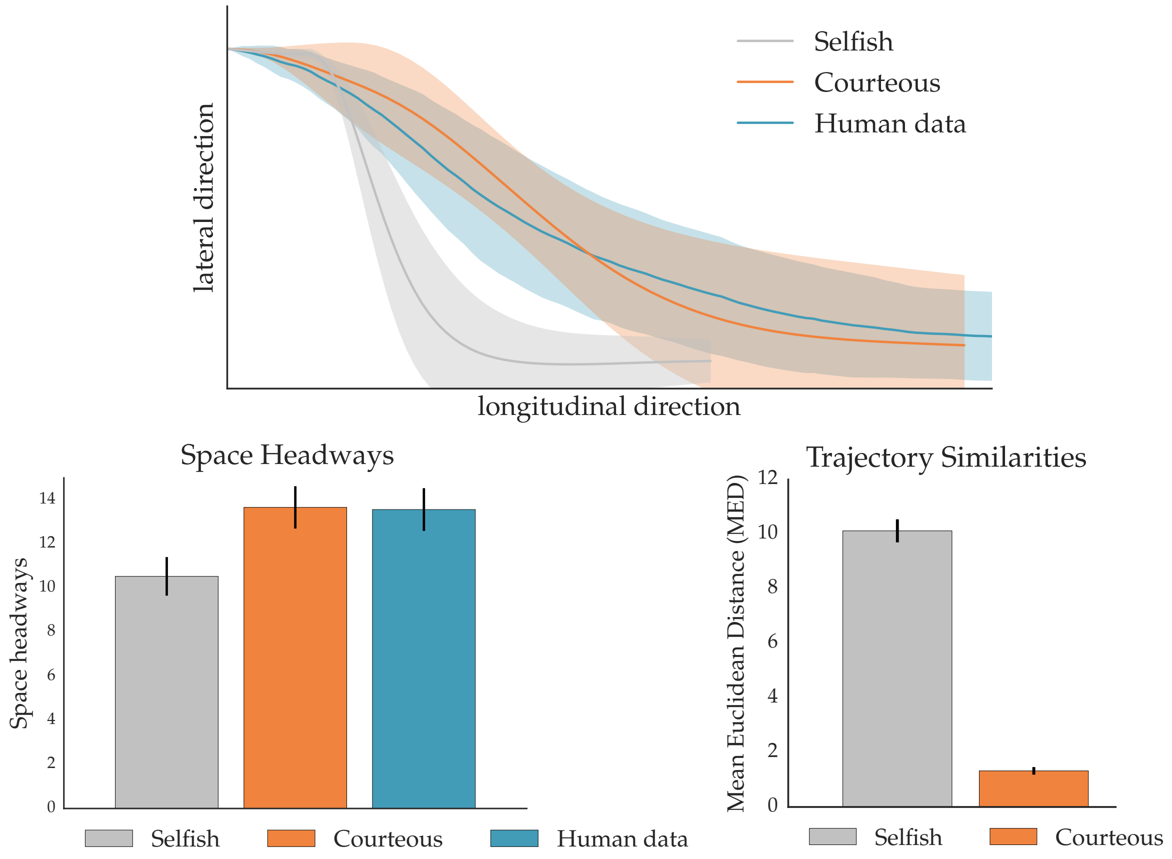


**Figure 3.12:** The courtesy term helps fit test set human driver data significantly better: we can see this from the actual trajectories (left), the following gaps (middle), and the mean euclidean distances from the ground truth human data (right).

## 3.6 Chapter Summary

As one of the key social compliance in human behavior, courtesy influences how humans interact with each other. In this chapter, we introduced our approaches to formulate such courtesy term and integrated it as a feature in the cost representation for human behavior. Utilizing autonomous vehicles as an application example, we designed a courteous planning strategy which encourages the autonomous vehicle to care not only its own utilities, but also the additional inconvenience it brings to by others. We saw that the introduction of the courtesy term not only can helps better capture real human driving behaviors, but also encourages the generation of more human-like robot behavior.

Like any research, the work in this chapter is limited in many ways. Human's behavior is diverse, and we studied only one aspect of it: the courtesy in interactions. More aspects such as the hierarchy and irrationality will be covered in later chapters. Another limitation is that we focused on two-player interaction where the robot was showing courtesy to a single human driver (we had other agents, but the robot did not attempt courtesy toward them). In real life, there will be many people on the road, and it becomes difficult to be courteous to all. Further work needs to push courtesy to the limits of interacting with multiple people in cases where it is difficult to be courteous to all.

Despite these limitations, we are encouraged to see that autonomous vehicles can generate more human-like behaviors via appropriate reward design learned from human demonstrations. We believe that the courteous planning strategy proposed in this chapter can be well applied to other application domains beyond autonomous driving, including for instance mobile robots in warehouses, home services robots and in general human-robot interactive scenarios.

# Chapter 4

# Hierarchical Reward Design and Learning

In Chapter 3, we formulated one aspect of the social compliance, i.e., the courtesy, to capture the human behavior and introduced it as a feature in the reward function for robots to encourage the generation of more human-like behavior. As mentioned in Chapter 1, beyond social compliance, human behavior is also naturally hierarchical [84, 109, 169, 120].

The work in this chapter discusses the problem of capturing the hierarchical structure in human's behavior policy and utilizing it in the reward design and learning. Again we use autonomous vehicles as an application example, but the approach developed can be well applied to other human-robot interaction systems such as UAVs, service robots, and mobile manipulators.

## 4.1 Introduction

Humans make decisions based on what they observe, what they believe, and what they want to achieve. Intrinsically, this is the human's behavior generation policy. Instead of being end-to-end, extensive evidence has been found, proving that *human's decision-making procedure is hierarchical*, particularly in complex scenarios [169, 120]. Imagine that you are driving on highways trying to change to the lane on your left where there is another car driving at a relatively lower speed, as shown in Fig. 4.1. The first *discrete decision* your probably will make is whether to change and enter the lane in front of the other car, or change from the behind. Once you make a decision, you will think about the *continuous* coordination between the lateral maneuvers and the longitudinal accelerations to execute that decision into trajectories. Similar scenarios happen at intersections where two cars arrive almost simultaneously. Before they drive through the intersection, they first need to decide whether to pass or yield. Once they decide to go, they will then pay more attention to how they accelerate.

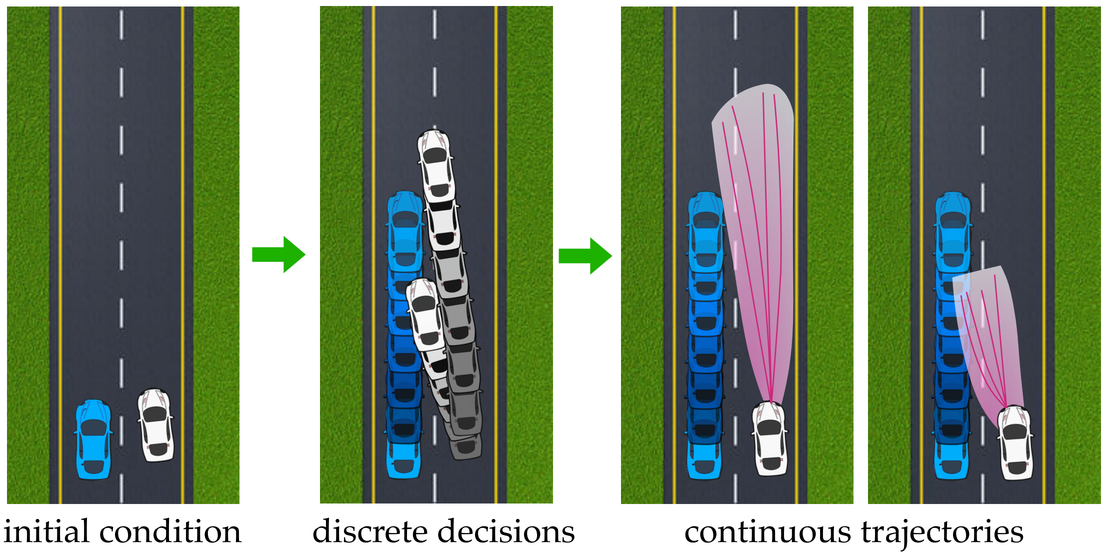


**Figure 4.1:** An illustration example on driving for the human's hierarchical decision-making procedure. A person (white) is trying to change to his/her left lane which is occupied by another human driver (blue)) at a relatively lower speed. First, the person should make a discrete decision about the way he/she change lanes, i.e., from the front or behind. Under each decision, he/she will decide how to maneuver the vehicle to fulfil the decision.

Actually, in most human-robot interaction systems, the hierarchical decision-making process exists, and there might be many layers involved. For instance, when a doctor tries to identify a disease, he might need to go through a decision tree to make a final judgement. Following our game-theoretic settings in Chapter 2, we focus on the scenarios with only two layers, with the first layer for discrete decisions and the second layer for continuous trajectories. The *discrete decisions* are also often interpreted as intentions or motion patterns in other literature [49] such as pass or yield, merge from the front or behind, and different driving styles for accelerations. The *continuous trajectories/maneuvers* are the temporal-spatial movements of the agent. As a hierarchical structure, discrete decisions from the first layer will influence the priors for the continuous trajectories. For instance, when a human decides to pass the intersection first while the other driver is waiting, he/she probably will pay less attention to the comfort. On the contrary, if he/she decides to pass after, he/she will accelerate more comfortably with smaller jerks.

*Our key observation in this chapter is that human's behavior generation policy is naturally hierarchical, involving both discrete and continuous interactive decisions. Such hierarchy and the internal influences should be reflected in the structure of our reward design to facilitate better description of human behavior.*

Contributions of this chapter are the following two.

**A formalism of the hierarchical reward design in a general human-robot interaction system.** We formalize the hierarchical reward function design in a general human-robot interaction system instead of cooperative scenarios as in [62]. By explicitly considering

the responses of one agent to another, we have translated the hierarchical reward design into a mixture of interation-aware behavior distribution of the human behavior. Both the future decisions and trajectories of the human depends not only on historical and current states, but also on potential plans of the interacting vehicle.
**A hierarchical inverse reinforcement learning framework on human driving data.** We explicitly consider the hierarchical planning procedure of human drivers, and formulate the influences of both discrete and continuous driving decisions. Via hierarchical IRL, the hierarchical reward for human drivers for highway merging is learned, which can be utilized for autonomous vehicles to generate either human-like merging actions or better prediction of human's merging behaviors.

## 4.2 Hierarchical Reward Design

In this section, we will autonomous vehicles to explain the design of the hierarchical reward design. We focus on the high-way merging scenario as shown in Fig. 4.2 where the white car is trying to merge onto the right-most lane of the highway occupied by the blue car who is doing lane keeping. Other cars (red) are treated as environment states and we do not model the interactions between them and our interaction system (white and blue cars).

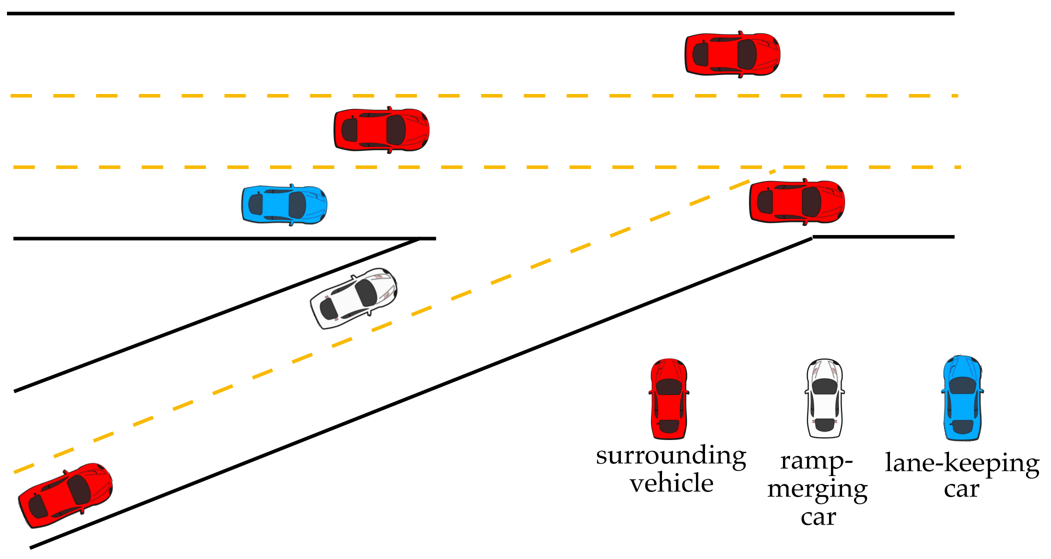


**Figure 4.2:** A ramp-merging scenario: we focus on the interaction between a ramp-merging car (white) and a lane-keeping car (blue). Other cars are treated as environment states.

We discuss the behavior for both the blue and white cares. When we design the cost functions for each of them, the one we are focusing on is defined as a target vehicle denoted by $(\cdot)_M$, and the other will serve as a host vehicle, denoted by $(\cdot)_H$. All other in-scene vehicles are treated as surrounding vehicles, denoted by $(\cdot)_O^i$ with $i=1,2,\cdots N$ where N is the number of surrounding vehicles. Throughout this chapter, we use $\xi$ to represent historical vehicle trajectories, and $\hat{\xi}$ for future trajectories. A trajectory is defined as a sequence of states, i.e., $\xi = [x_1^T, x_2^T, \cdots, x_L^T]^T$ where $x_i$ is the vehicle state at $i$-th time step. L is the trajectory length. Depending on the representation, the vehicle state can be different. For instance, it can simply be the positions of vehicles, or it can include velocities, yaw angles, accelerations, etc..

## 4.2.1 Features for Continuous Trajectories

The continuous trajectories describe the high-order dynamics of how human drive. The features we selected to parametrize the continuous trajectories can be grouped as follows:

- Speed - The incentive of the human driver to reach a certain speed limit $v_{\text{lim}}$ is captured by the feature

$$f_v(\hat{\xi}_M) = \sum_{t=0}^{L} (v_t - v_{\text{lim}})^2, \tag{4.1}$$

  where $v_t$ is the speed at time t along trajectory $\hat{\xi}_M$ and L is the length of the trajectory.

- Traffic - In dense traffic environment, human drivers tend to follow the traffic. Hence, we introduce a feature based on the intelligent driver model (IDM) [59]

$$f_{\text{IDM}}(\hat{\xi}_M) = \sum_{t=0}^{L} (s_t - s_t^{\text{IDM}})^2, \tag{4.2}$$

  where $s_t$ is the actual spatial headway between the front vehicle and predicted vehicle at time t along trajectory $\hat{\xi}_M$, and $s_t^{\text{IDM}}$ is the spatial headway suggested by IDM.

- Control effort and smoothness - Human drivers typically prefer to drive efficiently and smoothly, avoiding unnecessary accelerations and jerks. To address such preference, we introduce a set of kinematics-related features:

$$f_{\text{acc}}(\hat{\xi}_M) = \sum_{t=0}^{L} a_t^2, \quad f_{\text{jerk}}(\hat{\xi}_M) = \sum_{t=1}^{L} \left(\frac{a_t - a_{t-1}}{\triangle t}\right)^2, \tag{4.3}$$

  where $a_t$ represents the acceleration at time t along the trajectory $\hat{\xi}_M$. $\triangle t$ is the sampling time.

- Clearance to other road participants - Human drivers care about their distances to other road participants when they drive since distance is crucially related to safety. Hence, we introduce a distance-related feature

$$f_{\text{dist}}(\hat{\xi}_M) = \sum_{t=0}^{L} \sum_{k=1}^{N+1} e^{-\frac{(x_t - x_t^k)^2}{l^2} - \frac{(y_t - y_t^k)^2}{w^2}}, \tag{4.4}$$

  where $(x_t, y_t)$ and $(x_t^k, y_t^k)$ represent, respectively, the coordinates of the predicted vehicle along $\hat{\xi}_M$ and those of the k-th surrounding vehicle. Parameters $l$ and $w$ are the length and width of the predicted vehicle. We use coordinates in Frenet Frame to deal with curvy roads, i.e., $x$ denotes the travelled distance along the road and $y$ is the lateral deviation from the lane center.

- Goal - This feature describes the short-term goals of human drivers. Typically, goals are determined by the discrete driving decisions. For instance, if a lane-changing vehicle decides to merge in front of a host vehicle on his target lane, he will set his short-term goals to be ahead of the host vehicle. The goal-related feature is given by

$$f_g(\hat{\xi}_M) = \sum_{t=0}^{L} \|(x_t, y_t) - (x_t^g, y_t^g)\|_2^2. \tag{4.5}$$

- Courtesy - Most of human drivers view driving as a social behavior, meaning that they not only care about their own cost, but also care about others' cost, particularly when they are merging into others' lanes [135]. Suppose that the cost of the host vehicle is $C_H(\hat{\xi}_M, \xi, \hat{\xi}_H)$, then to address the influence of courtesy to the interaction, we introduce the feature

$$f_{\text{court}}(\hat{\xi}_M) = \max\left\{C_H\left(\hat{\xi}_M, \xi, \hat{\xi}_H\right) - C_H^{\text{default}}, 0\right\}. \tag{4.6}$$

  This feature describes the possible extra cost brought by the trajectory $\hat{\xi}_M$ of the predicted vehicle to the host vehicle, compared to the default host vehicle's cost $C_H^{\text{default}}$. We can learn about $C_H(\cdot)$ also from demonstrations. Details about this will be covered in the case study. For more details regarding the courtesy term, one can refer Chapter 3.

For vehicles in different driving scenarios, or under different discrete driving decisions, the features we used to parametrize their costs are different subsets of the above listed ones. For instance, drivers decided to merge behind or front would set different goals, and drivers with right of way would most likely care less about courtesy than those without.

*Remark I*: Note that all variables $v_t, a_t, \delta_t, x_t$ and $y_t$ in (4.1)-(4.6) can be expressed as functions of trajectory $\hat{\xi}_M$. For instance, if we define $\hat{\xi}_M = [x_0, y_0, \cdots, x_L, y_L]^T$ where $x$ and $y$ are the coordinates in Frenet Frame, then we can obtain all variables via numerical differentiation. Details are omitted.

### 4.2.2 Features for Discrete Decisions

Different from the continuous driving decisions that influence the higher-order dynamics of the trajectories, the discrete decisions determine the homotopy of the trajectories. To capture this, we selected the following two features to parametrize the cost function that induces the discrete decisions:

- Rotation angle - To describe the interactive driving behavior such as overtaking from left or right sides, merging in from front or back, we compute the rotation angle from $(x_t, y_t)$ to $(x_{t,H}, y_{t,H})$ along trajectory $\hat{\xi}_M \in \Xi_{d_M}$. Define the angle as $\omega_t$, and then the rotation angle feature is given by

$$f_{\angle}(d_M) = \sum_{t=0}^{L} \omega_t \tag{4.7}$$

  where $d_M$ is a discrete decision and L is the length of the trajectory $\hat{\xi}_M$.

- Minimum cost - It is also possible that human drivers make discrete decisions by evaluating the cost of trajectories under each decision, and select the one leading to the minimum-cost trajectory. To address this factor, we consider the feature

$$f_{\text{cost}}(d_M) = \min_{\tilde{\xi}_M} \boldsymbol{\theta}_{d_M}^T \mathbf{f}_{d_M}(\tilde{\xi}_M, \xi, \hat{\xi}_H). \tag{4.8}$$

  where $\boldsymbol{\theta}_{d_M}$ and $\mathbf{f}_{d_M}$, respectively, represent the learned parameters and selected features for the continuous trajectory distribution under discrete decision $d_M$.

## 4.3 Interaction-Aware Hierarchical Reward Learning

### 4.3.1 Hierarchical Rwards as a Mixture of Distributions

Since we present behavior generation policies via reward functions, different reward functions lead to different distributions of trajectories, as discussed in Chapter 2. Following this thread, in this section, we translate the learning of hierarchical reward function into a probabilistic prediction problem with a hierarchical structure.

Figure 4.3 illustrates such probabilistic and hierarchical trajectory–generation process for a lane-changing driving scenario. The predicted vehicle (blue) is trying to merge into the lane occupied by the host vehicle (red). Given all observed historical trajectories $\xi = \{\xi_O^{1:N}, \xi_H, \xi_M\}$ and his belief about the host vehicle's future trajectory $\hat{\xi}_H$, he first decides whether to merge behind the host vehicle ($d_M^1$) or merge front ($d_M^2$). Such discrete driving decisions are outcomes of the first-layer probability distribution $P(d_M | \xi, \hat{\xi})$, and partition the space of all possible trajectories into two distinct homotopy classes[2], each of

[2]Two trajectories belong to a same homotopy class if they can be continuously transformed to each other without collisions [62].

them can be described via a second-layer probability distribution $p(\hat{\xi}_M|d_M, \xi, \hat{\xi}_H)$. Note that among different homotopy classes, the distributions of the continuous trajectories can be significantly different since the driving preference under different discrete decisions may be different. For example, a vehicle decided to merge behind might care more about comfort and less about speed, while a vehicle merging front might care about completely the opposite.

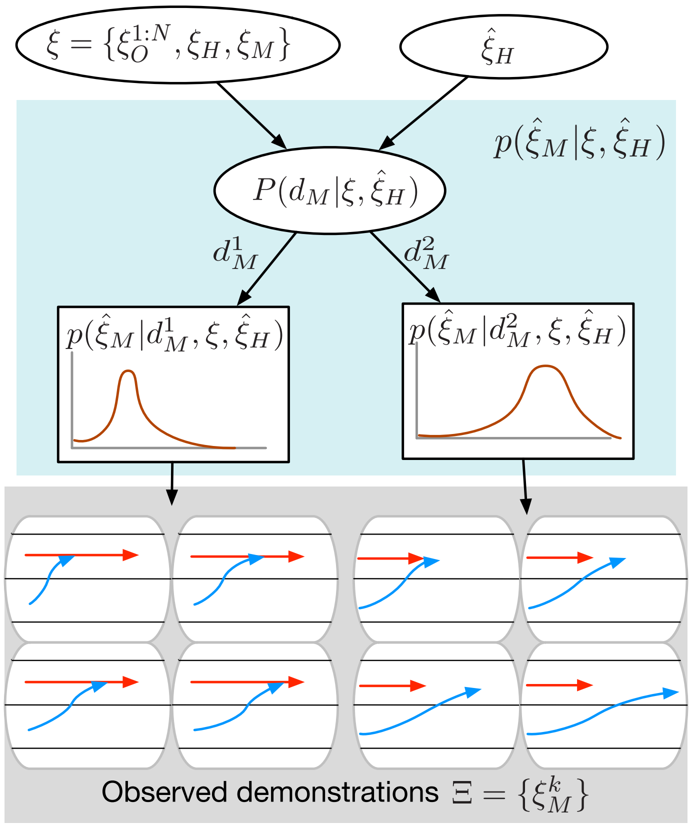

$\xi = \{\xi_O^{1:N}, \xi_H, \xi_M\}$
$\hat{\xi}_H$
$p(\hat{\xi}_M|\xi, \hat{\xi}_H)$
$P(d_M|\xi, \hat{\xi}_H)$
$d_M^1$
$d_M^2$
$p(\hat{\xi}_M|d_M^1, \xi, \hat{\xi}_H)$
$p(\hat{\xi}_M|d_M^2, \xi, \hat{\xi}_H)$
Observed demonstrations $\Xi = \{\xi_M^k\}$

**Figure 4.3:** The probabilistic and hierarchical trajectory generation process for a lane changing scenario. The predicted vehicle (blue) is trying to merge into the lane of the host vehicle (red). Given all observed historical trajectories $\xi$=$\{\xi_O^{1:N}, \xi_H, \xi_M\}$ and his belief about the host vehicle's future trajectory $\hat{\xi}_H$, the trajectory distribution of the predicted vehicle over all the trajectory space is partitioned first by the discrete decisions: merge behind ($d_M^1$) and merge front ($d_M^2$). Under different discrete decisions, the distributions of continuous trajectories can be significantly different, and each of them is represented via a probability distribution model. The observed demonstrations are samples satisfying the distributions.

Hence, the conditional distribution $p(\hat{\xi}_M|\xi, \hat{\xi}_H)$ is formulated as a mixture of distributions, which explicitly captures the influences of both discrete and continuous driving decisions of human drivers:

$$p(\hat{\xi}_M|\xi, \hat{\xi}_H) = \sum_{d_M^i \in \mathcal{D}_M} p(\hat{\xi}_M|d_M^i, \xi, \hat{\xi}_H) P(d_M^i|\xi, \hat{\xi}_H) \tag{4.9}$$

where $\mathcal{D}_M$ represents the set of all possible discrete decisions for the predicted vehicle. Again based on the principle of maximum entropy, we have the probability in each of the distribution on the right-hand side of (4.9) satisfying:

$$
\begin{aligned}
P(\hat{\xi}_M | d_M^i, \xi, \hat{\xi}_H; \boldsymbol{\theta}_{d_M}) &\propto e^{-C(\hat{\xi}_M, \xi, \hat{\xi}_H; \boldsymbol{\theta}_{d_M})} && (4.10)\\
P(d_M^i | \xi, \hat{\xi}_H; \mathbf{f}_d) &\propto e^{-C(d_M^i, \xi, \hat{\xi}_H; \mathbf{f}_d)} && (4.11)
\end{aligned}
$$

where $\boldsymbol{\theta}_{d_M}$ is the weight vector characterizing the cost function $C(\hat{\xi}_M, \xi, \hat{\xi}_H; \boldsymbol{\theta}_{d_M})$ for the continuous trajectory, while $\mathbf{f}_d)$ is the weight vector for the discrete cost function $C(d_M^i, \xi, \hat{\xi}_H; \mathbf{f}_d)$.

## 4.3.2 Learning of the Hierarchical Rewards

Suppose that the demonstration set $\Xi$ is partitioned into $|\mathcal{D}|$ subsets by the discrete decisions $d_M^i \in \mathcal{D}$. $|\mathcal{D}|$ is the dimension of $\mathcal{D}$. Each subset $\Xi_{d_M^i}$ contains trajectories that belong to the same homotopy class. We represent each demonstration in $\Xi_i$ by a tuple $(\hat{\xi}_M, d_M^i, \xi, \hat{\xi}_H)$ where $\xi = [\xi_O^{1:N}, \xi_H, \xi_M]$ represents all historical information.

### Continuous-Space IRL

Since the trajectory space is continuous and demonstrations are noisily local-optimal, we use Continuous Inverse Optimal Control with Locally Optimal Examples[74]. Under discrete decision $d_M$, we assume that the cost of each trajectory can be linearly parametrized by a group of selected features $\{\mathbf{f}_{d_M}\}$, i.e., $C(\boldsymbol{\theta}_{d_M}, \hat{\xi}_M, \xi, \hat{\xi}_H) = \boldsymbol{\theta}_{d_M}^T \mathbf{f}_{d_M}(\hat{\xi}_M, \xi, \hat{\xi}_H)$. Based on (4.10), the log likelihood of the given demonstration subset $\{\Xi_{d_M}\}$ is given by

$$
\log P(\Xi_{d_M} | \boldsymbol{\theta}_{d_M}) = \sum_{\hat{\xi}_M \in \Xi_{d_M}} \log \frac{e^{-C(\boldsymbol{\theta}_{d_M}, \hat{\xi}_M, \xi, \hat{\xi}_H)}}{\int e^{-C(\boldsymbol{\theta}_{d_M}, \tilde{\xi}_M, \xi, \hat{\xi}_H)} d\tilde{\xi}_M}. \tag{4.12}
$$

Our goal is to find the optimal $\boldsymbol{\theta}_{d_M}$ such that the given demonstration set is most likely to happen:

$$
\boldsymbol{\theta}_{d_M}^* = \arg\max_{\boldsymbol{\theta}_{d_M}} P(\Xi_{d_M} | \boldsymbol{\theta}_{d_M}). \tag{4.13}
$$

Using the Laplace approximation proposed in [74], we end up solving an optimization problem given by

$$
\min_{\boldsymbol{\theta}_{d_M}} \sum_{\hat{\xi}_M \in \Xi_{d_M}} g_{\hat{\xi}_M}^T(\boldsymbol{\theta}_{d_M}) H_{\hat{\xi}_M}^{-1}(\boldsymbol{\theta}_{d_M}) g_{\hat{\xi}_M}(\boldsymbol{\theta}_{d_M}) - \log |H_{\hat{\xi}_M}(\boldsymbol{\theta}_{d_M})|. \tag{4.14}
$$

Regarding more details, one can refer Section 2.5 in Chapter 2 or [74] as references.

### Discrete-Space IRL

Similarly, we assume that the decisions with lower cost are exponentially more probable. We also assume that the cost function is linearly parametrized by $\boldsymbol{\psi}$ and feature vector $\mathbf{f}^d = [f_\angle, f_{\text{cost}}]^T$, i.e., $C^d(d_M, \psi) = \boldsymbol{\psi}^T \mathbf{f}^d(d_M)$. Again, our goal is to find the optimal $\boldsymbol{\psi}^*$ such that the likelihood of the demonstration set $\Xi$ is maximized:

$$\max_{\boldsymbol{\psi}} P(\Xi|\boldsymbol{\psi}) = \max_{\boldsymbol{\psi}} \prod_{d_M \in \Xi} \frac{e^{-\boldsymbol{\psi}^T \mathbf{f}^d(d_M)}}{\sum_{\tilde{d}_M \in \mathcal{D}} e^{-\boldsymbol{\psi}^T \mathbf{f}^d(\tilde{d}_M)}}. \tag{4.15}$$

By taking the log probability and gradient-descent approach, the parameter $\boldsymbol{\psi}$ is updated via

$$\boldsymbol{\psi}_{s+1} = \boldsymbol{\psi}_s - \alpha \left( \frac{1}{N} \sum_{\Xi} \mathbf{f}^d(d_M) - \sum_{\tilde{d}_M} P(\tilde{d}_M|\boldsymbol{\psi}_s) f^d(\tilde{d}_M) \right), \tag{4.16}$$

$$P(\tilde{d}_M|\boldsymbol{\psi}_s) = \frac{e^{-\boldsymbol{\psi}_s^T \mathbf{f}^d(\tilde{d}_M)}}{\sum_{\tilde{d}_M \in \mathcal{D}} e^{-\boldsymbol{\psi}_s^T \mathbf{f}^d(\tilde{d}_M)}}, \tag{4.17}$$

where $\alpha$ is the step size and $N$ is the number of demonstrated trajectories in set $\Xi$. To estimate the probability $P(\tilde{d}_M|\boldsymbol{\psi}_s)$, we use sampling-based method. Detailed implementation will be covered later in the case study.

### Reverse Learning Process

As shown in Section 4.2.2, we utilize the reward function when acquiring the features for the discrete decisions. Hence, the reward learning process is performed in a reverse order: we first learn the continuous reward functions, utilize it to calculate the features for the discrete decisions, and then learn the cost function for the discrete decision layer.

## 4.4 A Case Study

In this section, we apply the proposed hierarchical IRL approach to model the interactive human driving behavior in a ramp-merging scenario.

### 4.4.1 Data Collection

We collect human driving data from the Next Generation SIMulation (NGSIM) dataset [6]. It captures the highway driving behaviors/trajectories by cameras mounted on top of surrounding buildings. The sampling time of the trajectories is $\triangle t$=0.01s. We choose 134 ramp-merging trajectories on Interstate 80 (near Emeryville, California), and separated them into two sets: a training set of size 80 (denoted by $\Xi$, i.e., the human demonstrations), and the other 54 trajectories as the test set.

Figure 4.4 shows the road map (See [6] for detailed geometry information) and an example group of trajectories. There are four cars in scene, one merging vehicle (red), one lane-keeping vehicle (blue) and two surrounding vehicles (black), with one ahead of the blue car and the other behind. Our interest focuses on the interactive driving behavior of both the merging vehicle and the lane-keeping vehicle.

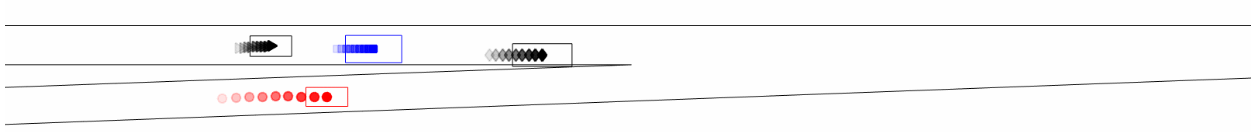


**Figure 4.4:** The merging map on Interstate 80 near Emeryville, California. Red: merging vehicle; Blue: lane-keeping vehicle; Black: other surrounding vehicles

## 4.4.2 Driving Decisions and Feature Selection

We use the same hierarchical IRL approach to model the conditional probability distributions for both the merging vehicle and the lane-keeping vehicle.

### Driving Decisions

In the ramp-merging scenario, the driving decisions are listed as in Table 4.1. As mentioned above, $\mathbf{x}=[x_1, \cdots, x_L]^T$ and $\mathbf{y}=[y_1, \cdots, y_L]^T$ are, respectively, the coordinate vectors in Frenet Frame along the longitudinal and lateral directions. L is set to be 50, i.e., in each demonstration, a trajectory with a length of 5s is collected.

| | Discrete Decisions | Continuous Decisions |
|---|---|---|
| merging-in vehicle | $\mathcal{D} =$ {merge front, merge back} | trajectory $\xi=[x_1, y_1, \cdots, x_L, y_L]^T$ |
| lane-keeping vehicle | $\mathcal{D} =$ {yield, pass} | trajectory $\xi=[x_1, y_1, \cdots, x_L, y_L]^T$ |

**Table 4.1:** Driving decisions for the interactive vehicles

### Feature Selection

Since the right of way for the merging vehicle and the lane-keeping vehicles are different, we define different features for them.

For the lane-keeping vehicle, the feature vectors related to the continuous driving decisions are as follows:

- Yield: $\mathbf{f}_{\text{yield}}=[f_v, f_{\text{acc}}, f_{\text{jerk}}, f_{\text{dist}}, f_g]^T$. We exclude the feature $f_{\text{IDM}}$ because once the lane-keeping driver decides to yield to the merging vehicle, it is very likely that he cares more about the relative positions to the merging vehicle instead of the heading space to the front vehicle when he plans his continuous trajectories. The goal position in $f_g$ is set to be $[x_{\text{current lane center}}, y_{t,\text{merging vehicle}} - s_0]$, i.e., $s_0$ behind the merging vehicle along longitudinal direction.

- Pass: $\mathbf{f}_{\text{pass}}=[f_v, f_{\text{IDM}}, f_{\text{acc}}, f_{\text{jerk}}, f_{\text{dist}}, f_g]^T$. In this case, the goal position in $f_g$ is set to be ahead of the merging vehicle along longitudinal direction, i.e., $[x_{\text{current lane center}}, y_{t,\text{merging vehicle}}+s_0]$. Also, if the driver decides to pass, it is more probable that the heading space to the front vehicle will influence the distribution of his continuous trajectories.

For the merging vehicle, the feature vectors for the continuous driving models are:

- Merge back: $\mathbf{f}_{\text{back}}=[f_v, f_{\text{acc}}, f_{\text{jerk}}, f_{\text{dist}}, f_g]^T$. The goal position is set to be $[x_{\text{target lane center}}, y_{t,\text{on-lane vehicle}}-s_0]$.
- Merge front: $\mathbf{f}_{\text{front}}=[f_v, f_{\text{IDM}}, f_{\text{acc}}, f_{\text{jerk}}, f_{\text{dist}}, f_{\text{court}}, f_g]^T$. Once the merging driver decides to merge in front of the lane-keeping vehicle, his heading space to the front lane-keeping vehicle is crucial to the distribution of his possible trajectories. Hence, we include feature $f_{\text{IDM}}$. Moreover, to respect the right of way of the lane-keeping vehicle, merging drivers might care about the extra cost they bring to the lane-keeping vehicle and prefer trajectories that induce less extra cost. To capture such effect, we add $f_{\text{court}}$. Given demonstrated trajectory group $\boldsymbol{\xi}=[\xi_{\text{merging}}, \xi_{\text{surroudings}}, \xi_{\text{lane-keeping}}]$, $f_{\text{court}}$ is computed via (4.6), i.e.,

$$f_{\text{court}}(\xi_{\text{merging}}) = C_{\text{lane-keeping}}(\boldsymbol{\theta}_{\text{yield}}, \boldsymbol{\xi}) - C^{\text{default}}_{\text{lane-keeping}} \tag{4.18}$$

  The cost function $C_{\text{lane-keeping}}(\boldsymbol{\theta}_{\text{yield}})=\boldsymbol{\theta}^T_{\text{yield}}\mathbf{f}_{\text{yield}}$ is learned with the features selected above. Regarding to $C^{\text{default}}_H$, we assume that the lane-keeping vehicle is by default following IDM.

### 4.4.3 Implementation Details and Training Performance

We use Tensorflow to implement the hierarchical IRL algorithm. Figure 4.5 gives the training curves regarding to both the continuous and discrete driving decisions. Due to the hierarchical structure, we first learn all four continuous distribution models for both the merging vehicle and the lane-keeping vehicle under different discrete decisions. We randomly sample subsets of trajectories from the training set and perform multiple trains. As seen from Fig. 4.5, the parameters in each continuous model converge quite consistently with small variance.

With the converged parameter vectors $\boldsymbol{\theta}_{\text{yield}}$, $\boldsymbol{\theta}_{\text{pass}}$, $\boldsymbol{\theta}_{\text{front}}$ and $\boldsymbol{\theta}_{\text{back}}$, the discrete feature vectors $\mathbf{f}^d$s are then calculated and thus the optimal parameter vectors $\boldsymbol{\psi}$s are learned via discrete IRL. To efficiently sample continuous trajectories under different discrete decisions, we first obtain the most-likely trajectories by optimizing the learned cost functions under each decision and then randomize them with additive Gaussian noise. The training curves (blue) are also shown in Fig. 4.5.

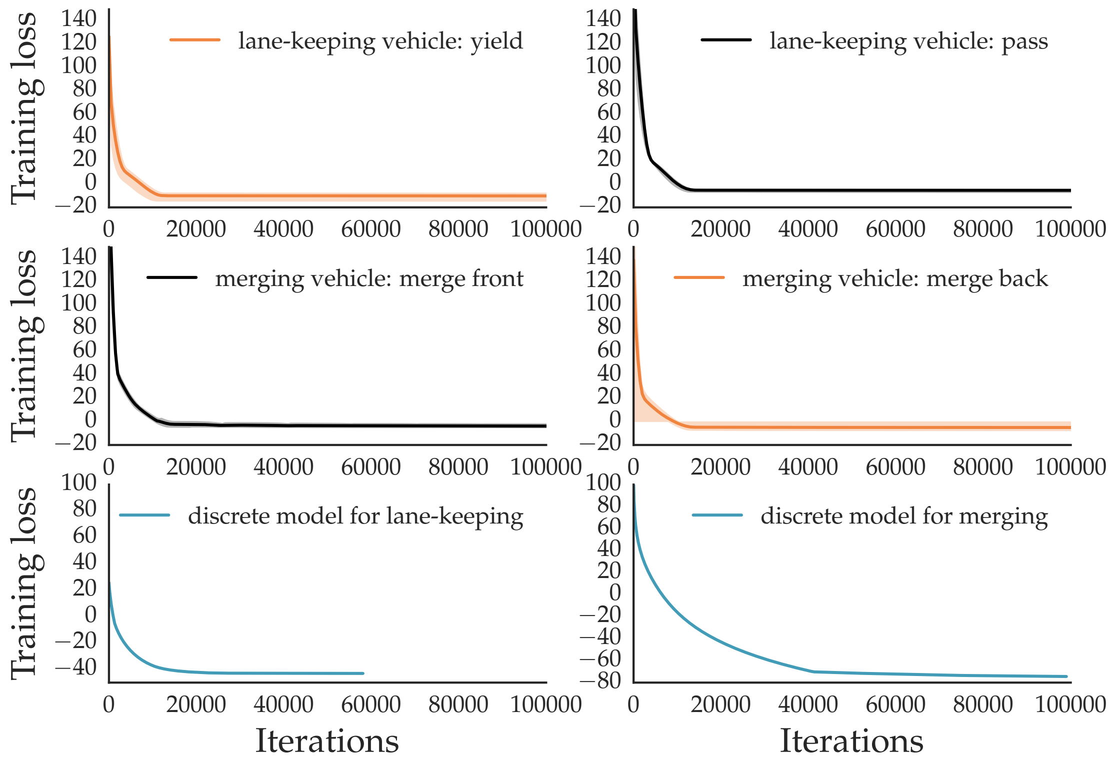


**Figure 4.5:** The training curves of both the lane-keeping vehicle and the merging vehicle under different discrete driving decisions

## 4.5 Test Results

Once $\boldsymbol{\theta}_{\text{yield}}$, $\boldsymbol{\theta}_{\text{pass}}$, $\boldsymbol{\theta}_{\text{front}}$, $\boldsymbol{\theta}_{\text{back}}$ and $\boldsymbol{\psi}_{\text{merging}}$, $\boldsymbol{\psi}_{\text{lane-keeping}}$ are acquired, we can obtain the conditional PDF defined in (4.9) via (4.9). With this PDF, probabilistic and interactive prediction of human drivers' behavior can be obtained.

We perform two different tests on our test set to evaluate the accuracy of the learned probabilistic model.

### 4.5.1 Accuracy of the probabilistic prediction of discrete decisions

To measure the accuracy of the probabilistic prediction of discrete decisions, we extract N=2000 short trajectories in the test set with a horizon length of 10 from the 54 long trajectories. For each short trajectory, starting from the same initial condition, we sample $M=4$ trajectories with different motion patterns (discrete decisions) in the spatiotemporal domain. One of them is set to be the same as ground truth.

**Metric:** We adopt a fatality-aware Brier metric [21] and the fatality-aware metric in [161] to evaluate the prediction accuracy. Brier score calculates the difference between ground-truth probabilities and the predicted ones via

$$\mathcal{B} = \frac{1}{NM} \sum_{i}^{N} \sum_{j}^{M} \left(P_{i,j} - O_{i,j}\right)^2 \tag{4.19}$$

where $P_{i,j}$ and $O_{i,j}$ represent the predicted probability and ground-truth probability for sampled trajectory $(i, j)$, respectively. The fatality-aware metric in [161] refines the Brier score by three different aspects: ground-truth accuracy ($\mathcal{G}$) measuring the prediction error of the ground-truth motion patterns, conservatism accuracy ($\mathcal{C}$) measuring the false alarm of aggressive motion patterns, and non-defensiveness accuracy ($\mathcal{D}$) measuring the miss detection of dangerous motion patterns. One can refer to [161] for more details.

**Comparison:** We compare the prediction among three different approaches: the proposed hierarchical IRL method, a neural-network (NN) based method [16] and a hidden markov models (HMM) [111] based method.

**Results:** The scores for all three methods are shown in Table 4.2. We can see that the proposed method can yield better prediction performance than HMM based method, and similar accuracy with the NN based method.

| | HMM | NN | Hierarchical IRL |
|---|---|---|---|
| $\mathcal{B}$ | 0.1043 | 0.1099 | 0.1821 |
| $\mathcal{G}$ | 0.0701 | 0.0563 | 0.1117 |
| $\mathcal{C}$ | 0.0476 | 0.0493 | 0.0178 |
| $\mathcal{D}$ | 0.1356 | 0.1303 | 0.0698 |
| $\mathcal{B}_c = \mathcal{G} + \mathcal{C} + \mathcal{D}$ | 0.2551 | 0.2361 | 0.2053 |

**Table 4.2:** Scores of different probabilistic prediction approaches

### 4.5.2 Accuracy of the probabilistic prediction of continuous trajectories

In this test, we generate the most probable trajectories under different discrete driving decisions by solving a finite horizon Model Predictive Control (MPC) problem using the above learned continuous cost functions. We show three illustrative examples in Fig. 4.6. The red dotted lines and blue solid lines represent, respectively, the predicted most-likely trajectories and the ground truth trajectories. The thick black dash-dot lines are trajectories of other vehicles. We can see that the predicted trajectories are very close to the ground truth ones.

To quantitatively evaluate the accuracy of the prediction, we calculate the trajectory similarities between the predicted one and the ground truth in terms of Mean Euclidean Distance (MED) [110].

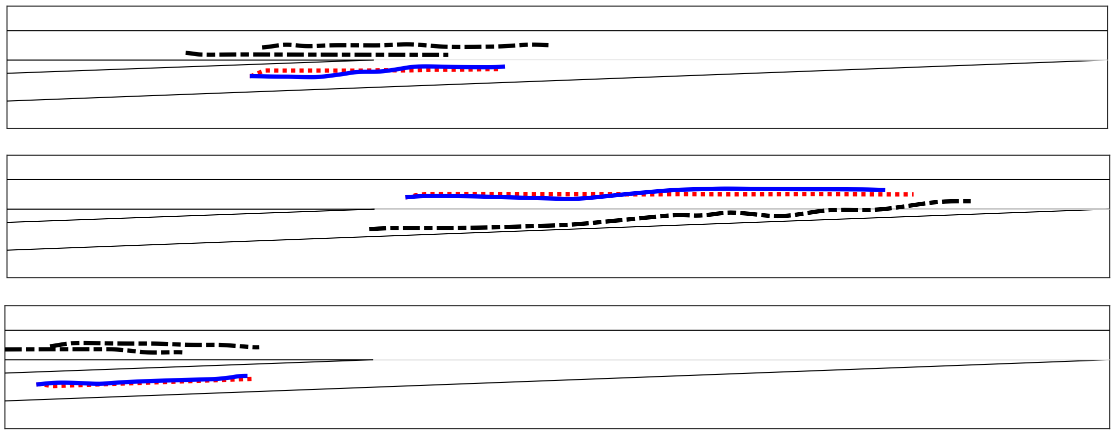


**Figure 4.6:** Three illustrative examples of the re-generated most probable trajectories (red dotted line) compared with the ground truth trajectories (blue solid line). Thick black dash-dot lines represent the trajectories of other vehicles except for the predicted one.

**Metric:** We adopt Mean Euclidean Distance (MED) for trajectories. Given the ground truth trajectory $\xi_{\text{ground}}=[x_{g,1}, y_{g,1}, \cdots, x_{g,L}, y_{g,L}]^T$ and predicted trajectory $\xi_{\text{prediction}} = [x_{p,1}, y_{p,1}, \cdots, x_{p,L}, y_{p,L}]^T$ of same length $L$ and same sampling time $\triangle T$, the trajectory similarity is calculated as follows:

$$\mathcal{S}_{\text{MED}} = \frac{1}{L}\sum_{i=1}^{L} \| \left[x_{p,i}, y_{p,i}\right]^T - \left[x_{g,i}, y_{g,i}\right]^T \|_2 \tag{4.20}$$

**Results:** We test on 20 long trajectories in the test set, and results are summarized in Table 4.3: the proposed hierarchical IRL method can achieve trajectory prediction with a mean MED of 0.6172m with standard deviation of 0.2473m.

| | Mean (m) | Max (m) | Min (m) | Std (m) |
|---|---|---|---|---|
| MED | 0.6172 | 1.0146 | 0.3644 | 0.2473 |

**Table 4.3:** Trajectory similarities in terms of MED

## 4.6 Chapter Summary

In this chapter, we discussed the hierarchical reward design and learning to better describe the human behavior. Motivated by the naturally hierarchical mechanism in human's decision-making process, we have formulated a two-layer hierarchical reward structure. With the principle of maximum entropy, such hierarchy in reward functions translates into a mixture of distributions. Hence, the learning of the hierarchical rewards was cast into an interaction-aware probabilistic prediction problem. We applied the proposed method on a ramp–merging driving scenario with data collected from the NGSIM dataset. The quantitative results verified the effectiveness of the proposed approach in terms of accurate representation and re-generation to both discrete decisions and continuous trajectories.

As always, the work in this chapter also has many limitations. For instance, we only focused on two-layer hierarchical decision making process of human. In practice, however, there might be many more layers and the mutual influence among layers can be more complicated. Another limitation lies in the design of feature sets in both the discrete and continuous spaces. The performance of the proposed method depends on appropriate selection of the features. Some systematic feature construction framework is desired to facilitate this process.

Despite of these limitations, combined with Chapter 3, the work in this chapter has brought another aspect into the reward design to represent human behavior: the hierarchy. We are excited to see that such structure indeed facilitate better description of human behavior. Moreover, compared to end-to-end learning, the hierarchy provides much more interpretability of the human's policy, which is easier to use in optimization-based robotic behavior generation. It will help increase the transparency of our robotic behavior in human-robot interaction which is of significant importance for the wide adoption of intelligent autonomous systems.

# Chapter 5

# Irrationality-Compatible Reward Design and Learning

In this chapter, we will address another aspect of the human behavior: the irrationality. The motivation for this chapter comes from one remaining challenge for the utility-driven human models. That is most of existing them utility-driven human models assume rationality (or at least noisy rationality) of human with respect to the expected utility theory (EUT). However, there have been substantial evidences in various domains contradicting such assumption. Human behavior is often found to be systematically deviating from the optimal (or rational) behavior predicted by EUT.

Hence, in this chapter, we propose an interpretable human behavior model in interactive scenarios based on the cumulative prospect theory (CPT). As a non-expected utility theory, CPT can well explain some systematically biased or "irrational" behavior/decisions of human that cannot be explained by the expected utility theory. We present our design on autonomous vehicles, but again the proposed framework can be applied to other human-robot interaction systems.

## 5.1 Introduction

In the past decade, a great amount of effort has been devoted to behavior modelling of human, e.g., [3, 149, 112]. Most of the proposed methodologies can be categorized into three groups: 1) predefined models, 2) learning models, and 3) utility-driven models. Predefined models generate driving behavior based on IF-THEN rules [87], or selected key indices such as time-to-collision (TTC) and time-to-intersection (TTI) [61], or some analytical functions dedicated to describe the behavior in specific scenarios. Examples include the intelligent driver model (IDM) [145] for car following and the minimizing overall braking model for lane changing (MOBIL) [60]. Such predefined models are highly interpretable, i.e., explicit physical meanings can be found for all the model structures, variables and parameters. However, these models typically require lots of manual work in designing structures and tuning parameters, which can be overwhelming tasks when the amount of data is large.

Learning models generate driving behavior based on trained machine-learning models. They can be either discriminative models such as support vector machines (SVM) [9] and mixture density network (MDN) [49], or generative models such as hidden Markov models (HMM) [75], generative adversarial networks (GAN) [76, 42] and variational auto-encoder (VAE) [48, 85, 50]. Compared to the predefined models, such learning models can better approximate the complicated distributions of human behavior in massive driving data without manual tuning of model parameters. However, they also suffer from several fundamental problems. First, most of the learning models, particularly the deep networks, are data-hungry. With a relatively small amount of data, they can hardly achieve satisfactory performance. Even with sufficient amount of data supplied, they still suffer from a second problem: the lack of causality and interpretability of the learned behavior model. Consequently, it is hard to efficiently generalize them to new scenarios such as those with a varying number of agents or new driving maps.

Utility-driven models come from the theory of mind (TOM) [108]. A key feature of these models is that they leverage the fact that human drivers are not random agents, but agents who optimize some utility functions. Hence, they assume that human drivers try to make decisions or plan trajectories that maximize their utilities (or minimize the costs). Such assumption is often known as the Boltzmann noisily rational model [14]. Stemming from TOM, utility-driven models provide causality inherently, and are more interpretable since all the utilities and constraints are associated with explicit physical meanings, as mentioned in [31]. In order to infer the various utility functions of human from actual driving data, inverse optimal control (or inverse reinforcement learning (IRL)) [1, 166, 73] has been well adopted. For instance, [135] use IRL to model the courteous behavior. [132] and [47] use it for probabilistic reaction prediction under social interactions. Benefiting from causal and interpretable structure design, the utility-driven models are more data-efficient, i.e., a satisfactory model can be learned with a relatively small amount of data. Hence, they provide a promising balance between interpretability, model flexibility, and data-efficiency.

One remaining challenge for the utility-driven models is that, as mentioned above, most of them assume the rationality (or at least noisy rationality) of human drivers with respect to the expected utility theory (EUT). However, there have been substantial evidences in various domains contradicting such assumption. Human behavior is often found to be systematically deviating from the optimal (or rational) behavior predicted by EUT. Examples can be found as framing effect, risk-seeking behavior, loss-aversion behavior, and so on [55, 146]. In this chapter, we define such systematically biased behavior from EUT as "*irrational behavior*." In driving scenarios, such irrational behavior can be well observed, particularly when the drivers are interacting with each other. Under such circumstances, the traditional EUT-based utility-driven models can no longer correctly predict the human behaviors, which might cause collisions for the autonomous vehicles.

*We aim to extend the utility-based behavior generation model to capture both the rational and irrational behavior of human drivers.*

Towards this goal, we reformulate the utility-based models in the framework of the cumulative prospect theory (CPT) [146] - a well-known non-expected utility theory (NEUT) that can explain many of the irrational behaviors mentioned above. Afterwards, a hierarchical learning algorithm is proposed to learn the utility function, the value function, and the decision weighting function in the developed CPT model. A case study for roundabout merging is presented with real data from the INTERACTION dataset [158, 159]. Prediction performances of three different models are compared: a predefined model based on TTC, a learning-based model based on a neural network, and the proposed CPT-based model. The results show that the proposed model outperforms the TTC model, and achieves similar performance as the learning-based method with much less training data and better interpretability.

## 5.2 Modelling the Decision-Making Process via CPT

### 5.2.1 Extension from EUT to CPT

As discussed in Section 2.3.2 and Section 2.3.3 in Chapter 2, CPT is proposed to explain the irrational behaviors of human beyond the scope of EUT. We briefly review the decision networks for both EUT and CPT. For more details, one can refer Section 2.3.2 and Section 2.3.3 in Chapter 2.

Let $a$ denote a decision, and $x$ denote a state or an event. Given the action $a$, the probability for the occurrence of $x$ is represented by $p(x|a)$. The utility of the action-state pair is $u(a, x)$. Then as shown in Fig. 5.1, the expected utility is given by $\sum_x p(x|a)u(a, x)$. A rational agent under this measure will try to find the action $a^*$ which maximizes the expected utility, i.e., $a^* = \arg\max \sum_x p(x|a)u(a, x)$.

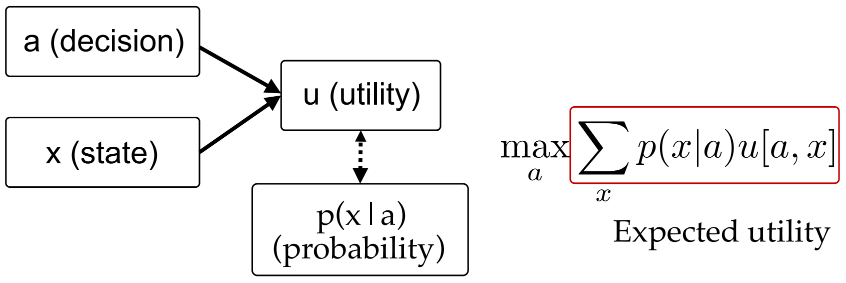


**Figure 5.1:** The decision network with EUT as a measure: a rational agent tries to find the action $a^*$ which maximizes the expected utility, i.e., $a^* = \arg\max \sum_x p(x|a)u(a, x)$.

Figure 5.2 describes the decision network under the CPT measure. Different from the EUT, optimal agents under this measure does not choose decisions that maximize the expected utilities, but rather they prefer the decisions that maximize the cumulative prospect. As shown in Fig. 5.2, a cumulative prospect is defined as

$$V(a) = \sum_{x} \pi^{+}\left(w_{\text{cum}}^{+}(p(x|a))\right) v^{+}(u[a,x]) + \pi^{-}\left(w_{\text{cum}}^{-}(p(x|a))\right) v^{-}(u[a,x]) . \tag{5.1}$$

where the definitions of $\pi^{\pm}$, $w_{\text{cum}}^{\pm}$ and $v^{\pm}$ are given in (). Compared to EUT in Fig. 5.1, CPT has more flexibilities by incorporating three more elements:

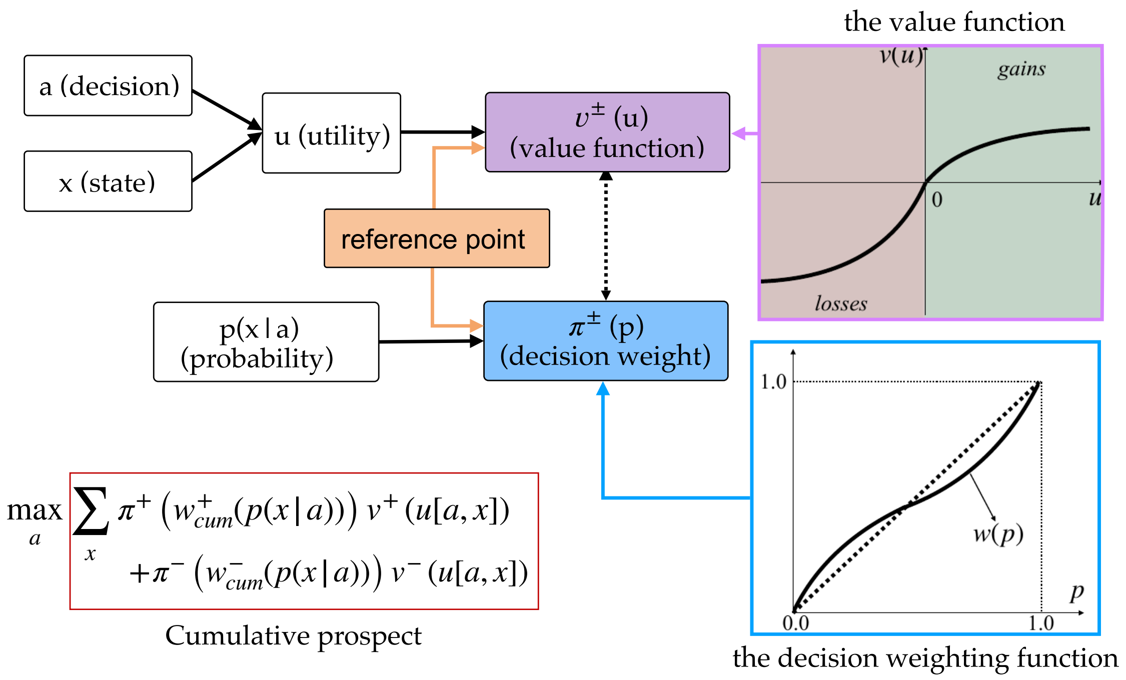

a (decision)
x (state)
u (utility)
$v^{\pm}$ (u) (value function)
reference point
p(x | a) (probability)
$\pi^{\pm}$ (p) (decision weight)
the value function
$v(u)$
gains
losses
0
$u$
1.0
0.0
1.0
$w(p)$
$p$
the decision weighting function
$\max_a \sum_x \pi^+ \left(w^+_{cum}(p(x\,|\,a))\right) v^+(u[a,x]) + \pi^- \left(w^-_{cum}(p(x\,|\,a))\right) v^-(u[a,x])$
Cumulative prospect

**Figure 5.2:** The decision network with CPT as a measure: optimal agents does not choose decisions that maximize the expected utilities, but rather they prefer the decisions that maximize the cumulative prospect defined in (5.1).

- Reference point: a reference point defines a reference utility in the decision network. When human make decisions, instead of evaluating the absolute utilities, human compare the utilities to the reference point and evaluate the utilities of actions as gains and loss which are denoted by $(\cdot)^{+}$ and $(\cdot)^{-}$, respectively.
- Value function: for both gains and loss, human further evaluate the utilities via another set of value functions $v^{\pm}$. The value functions are typically concave for gains and convex for loss, as shown in Fig. 5.2.

- Decision weighting function: instead of weighting the influence of different utilities by their probabilities as in EUT, CPT uses another set of functions, the cumulative decision weighting functions $\pi^{\pm}\left(w^{\{}\pm\}_{\text{cum}}\right)$. As shown in Fig. 5.2, the decision weighting function is of S-shape. It can help explain human's irrationality in terms of their underestimation of high-probability events and overestimation of low-probability ones.

Many experiment studies have showed that representative functional forms for $v$ and $w$ can be written as

$$\begin{aligned} v(u) &= \begin{cases} (u-u_0)^{\alpha}, & \text{if } u \geqslant u_0 \\ -\lambda(u_0-u)^{\beta}, & \text{if } u < u_0 \end{cases} && (5.2) \\ w^{+}(p) &= \frac{p^{\gamma}}{(p^{\gamma}+(1-p)^{\gamma})^{1/\gamma}}, && (5.3) \\ w^{-}(p) &= \frac{p^{\delta}}{(p^{\delta}+(1-p)^{\delta})^{1/\delta}}. && (5.4) \end{aligned}$$

where $\alpha, \beta, \gamma, \delta \in (0,1]$ and $\lambda \geqslant 1$. If there are $m$ possible events/states, the cumulative functions $\pi^{\pm}$ for each event are defined as

$$\begin{aligned} \pi_m^{+} &= w^{+}\left(p(x_m)\right), \quad \pi_m^{-} = w^{-}\left(p(x_m)\right), && (5.5) \\ \pi_j^{+} &= w^{+}\left(\sum_{k=j}^{m} p(x_k)\right) - w^{+}\left(\sum_{k=j+1}^{m} p(x_k)\right), && (5.6) \\ \pi_j^{-} &= w^{-}\left(\sum_{k=j}^{m} p(x_k)\right) - w^{-}\left(\sum_{k=j+1}^{m} p(x_k)\right), \quad \forall j=1,\cdots.m-1 && (5.7) \end{aligned}$$

### 5.2.2 Behavior Modeling via CPT

We consider the driving scenarios with two interacting drivers. Each driver has two discrete decisions/actions: yield and pass. Such scenario can be found in many urban driving circumstances such as intersections, roundabouts and ramp merging.

Throughout the chapter, we refer the predicted vehicle as the target vehicle (denoted with subscript $(\cdot)_T$), and the other one as the interacting vehicle (with $(\cdot)_I$). Denote the action set with pass and yield as $\{\boldsymbol{a}\}=\{a_p, a_y\}$. At time $t$, given the historical trajectories of both vehicles, $\{\xi_I^t, \xi_T^t\}$, we aim to obtain an interpretable decision-making model to predict the decision of the target vehicle. Note that in interactive driving scenarios, the responses from the interacting vehicle are probabilistic in nature, which will bring uncertainties to the decision-making process of the target vehicles. Under the decision $a_p$, the target vehicle has to consider the possibility of the interacting vehicle not yielding, which might force the target vehicle to brake and fail to pass. For the decision $a_y$, however, we can assume that it will always succeed. Hence, the prospects for actions $a_p$ and $a_y$ are, respectively,

$$\begin{aligned} P_{a_p} &= \left\{\left(u(\hat{\xi}_{I,y},\hat{\xi}_{T,p}),p_{I,y}\right),\left(u(\hat{\xi}_{I,ny},\hat{\xi}_{T,p}),1-p_{I,y}\right)\right\}, && (5.8)\\ P_{a_y} &= \left\{\left(u(\hat{\xi}_{I,ny},\hat{\xi}_{T,y}),1.0\right)\right\}. && (5.9)\end{aligned}$$

where $p_{I,y}$ represents the probability of the interacting vehicle yielding to the target one, and $\hat{\xi}_{I,y}$ and $\hat{\xi}_{I,ny}$ are, respectively, the yielding and non-yielding trajectories of the interacting vehicle. Similarly, $\hat{\xi}_{T,p}$ and $\hat{\xi}_{T,y}$ are, respectively, the passing and yielding trajectories of the target vehicle. $u(\hat{\xi}_{I,y},\hat{\xi}_{T,p})$ is the utility for the interacting vehicle if it takes a yielding trajectory $\hat{\xi}_{I,y}$ and the target vehicle takes a passing trajectory $\hat{\xi}_{T,p}$. Similarly, $u(\hat{\xi}_{I,ny},\hat{\xi}_{T,p})$ is the utility of the interacting vehicle if it takes non-yielding trajectory $\hat{\xi}_{I,y}$ but the target vehicle takes a passing trajectory $\hat{\xi}_{T,p}$. The last term $u(\hat{\xi}_{I,ny},\hat{\xi}_{T,y})$ defines the case when the interacting vehicle takes a non-yielding trajectory $\hat{\xi}_{I,ny}$ and the target vehicle yields with $\hat{\xi}_{T,y}$.

Set $u_0$=0. Recalling the CPT model defined in (5.1)-(5.4), we can write the CPT values of the target vehicle under different decisions as:

$$\begin{aligned} V(a_p) &= v\left(u^+(\hat{\xi}_{I,y},\hat{\xi}_{T,p})\right)\left(w^+(1.0)-w^+(p_{I,y})\right)\\ &\quad +v\left(u^+(\hat{\xi}_{I,ny},\hat{\xi}_{T,p})\right)w^+(p_{I,y})\\ &= \left(u(\hat{\xi}_{I,y},\hat{\xi}_{T,p})\right)^\alpha\left(1-\frac{p_{I,y}^\gamma}{\left(p_{I,y}^\gamma+(1-p_{I,y})^\gamma\right)^{1/\gamma}}\right)\\ &\quad +\left(u(\hat{\xi}_{I,ny},\hat{\xi}_{T,p})\right)^\alpha\frac{p_{I,y}^\gamma}{\left(p_{I,y}^\gamma+(1-p_{I,y})^\gamma\right)^{1/\gamma}}, && (5.10)\\ V(a_y) &= v\left(u^+(\hat{\xi}_{I,ny},\hat{\xi}_{T,y})\right)w^+(1.0)=\left(u(\hat{\xi}_{I,ny},\hat{\xi}_{T,y})\right)^\alpha. && (5.11)\end{aligned}$$

Note that in (5.10)-(5.11), $u_0$=0 simplifies $u(\cdot)$ to $u^+(\cdot)$.

The decision of the target vehicle is then written as

$$a^* = \arg\max_{a\in\{a_p,a_y\}}\{V(a_p),V(a_y)\}. \tag{5.12}$$

## 5.3 Hierarchical Learning of CPT-based Reward

In the CPT-based decision-making model given in (5.10)-(5.11), we have many unknowns that need to be learned from data: the parameters $\alpha$ and $\gamma$, the utility function

$u(\cdot)$, and the probability $p_{I,y}$ given $\{\xi_I^t, \xi_T^t\}$. We propose to learn them hierarchically based on the following two assumptions:
**Assumption** 1: The parametrization of the utility function of the target vehicle $u$:$(\xi_I, \xi_T) \to \mathcal{R}$ does not change with decisions. For instance, if we assume that $u$ is a linear combination of a set of features defined on trajectories, the weights of the features will not change.
**Assumption** 2: When the target vehicle is evaluating the CPT value under each decision, the best achievable utilities corresponding to different responses of the interacting vehicle will be adopted. Namely, in (5.10)-(5.11), we assume $u(\hat{\xi}_{I,y}, \hat{\xi}_{T,p}) \approx u(\hat{\xi}_{I,y}, \xi^*_{T,p}(\hat{\xi}_{I,y}))$, $u(\hat{\xi}_{I,ny}, \hat{\xi}_{T,p}) \approx u(\hat{\xi}_{I,ny}, \xi^*_{T,p}(\hat{\xi}_{I,ny}))$, and $u(\hat{\xi}_{I,ny}, \hat{\xi}_{T,y}) \approx u(\hat{\xi}_{I,ny}, \xi^*_{T,y}(\hat{\xi}_{I,ny}))$.

## 5.3.1 Learning the utility function

We start with learning the utility function $u$ for the target vehicle. With Assumption 1, we learn $u$ from a set of decision-free trajectories of the target vehicle, so that influences of decisions on the demonstrated trajectories can be avoided. This transforms the learning of the utility function into a typical IRL problem. We assume that $u$ is a linear combination of a set of selected features $\boldsymbol{\phi} = [\phi_1, \phi_2, \cdots, \phi_M]$ defined over $(\xi_I^{1:N}, \xi_T^{1:N})$ with a horizon length of $N$:

$$u(\xi_I^{1:N}, \xi_T^{1:N}; \boldsymbol{\theta}) \quad = \quad \boldsymbol{\theta}^T \sum_{k=1}^{N} \boldsymbol{\phi}(\xi_I^k, \xi_T^k). \tag{5.13}$$

The goal is to find the weights $\boldsymbol{\theta}$ which maximizes the likelihood of the demonstration set $\mathcal{U}_D$:=$\{(\xi_{I,i}^{1:N}, \xi_{T,i}^{1:N}), i{=}1, \cdots, |\mathcal{U}_D|\}$:

$$\theta^* = \arg\max_{\boldsymbol{\theta}} P(\mathcal{U}_D|\boldsymbol{\theta}). \tag{5.14}$$

With the principle of maximum entropy [166], trajectories with higher utilities are exponentially more likely:

$$P(\xi_{T,i}^{1:N}, \boldsymbol{\theta}|\xi_{I,i}^{1:N}) \propto \exp\left(u(\xi_{T,i}^{1:N}, \xi_{I,i}^{1:N}; \boldsymbol{\theta})\right). \tag{5.15}$$

Thus (6.13) becomes

$$\begin{aligned} \boldsymbol{\theta}^* \quad &= \quad \arg\max_{\boldsymbol{\theta}} P(\mathcal{U}_D|\boldsymbol{\theta}) {=} \arg\max_{\boldsymbol{\theta}} \Pi_{i=1}^{|\mathcal{U}_D|} \frac{P(\xi_{T,i}^{1:N}, \boldsymbol{\theta}|\xi_{I,i}^{1:N})}{P(\boldsymbol{\theta})} \\ &= \quad \arg\max_{\boldsymbol{\theta}} \Pi_{i=1}^{|\mathcal{U}_D|} \frac{P(\xi_{T,i}^{1:N}, \boldsymbol{\theta}|\xi_{I,i}^{1:N})}{\int P(\tilde{\xi}_T^{1:N}, \boldsymbol{\theta}|\xi_{I,i}^{1:N}) d\tilde{\xi}_T^{1:N}}. \end{aligned} \tag{5.16}$$

To solve (6.15), we use the continuous-domain IRL algorithm proposed in [73]. One can refer Section 2.5 in Chapter 2 or [73] for details.

### 5.3.2 Evaluating the Utilities and Probabilities

Once the utility function $u$ is obtained, we can generate the best achievable utilities under different decisions. Based on Assumption 2, utilities under different decisions are generated as follows:

- $u(\hat{\xi}_{I,y}, \hat{\xi}_{T,p})$ describes the utility when the target vehicle passes, and the interacting vehicle yields. It can be approximated by $u(\hat{\xi}_{I,y}, \xi^*_{T,p}(\hat{\xi}_{I,y}))$ with $\xi^*_{T,p}(\hat{\xi}_{I,y}) = \text{argmax}_{\xi_{T,p}} u\left(\hat{\xi}_{I,y}, \xi_{T,p}\right)$. Intuitively, this utility is equivalent to the best achievable utility as if the interacting vehicle was not there since it would yield to the target vehicle.

- $u(\hat{\xi}_{I,ny}, \hat{\xi}_{T,p})$, on the other hand, describes the utility when the target vehicle passes but with a non-yielding interacting vehicle. Under this situation, the target vehicle might have to brake and terminate the action of passing. Therefore, we set $\hat{\xi}_{I,ny}$ as if the interacting vehicle was maintaining its initial speed. $\xi^*_{T,p}(\hat{\xi}_{I,ny})$ is calculated as $\xi^*_{T,p}(\hat{\xi}_{I,ny}) = [\xi^{*,1:k_0}_{T,p}(\hat{\xi}_{I,y}); \xi^{k_0:N}_{T,brake}]$ where the first part $\xi^{*,1:k_0}_{T,p}(\hat{\xi}_{I,y})$ is the first $k_0$ steps of an optimal passing trajectory, while the second part $\xi^{k_0:N}_{T,brake}$ is a braking trajectory in order to avoid collision with the interacting vehicle. The maximum value of $k_0$ is found via boundaries on deceleration. Hence, the corresponding utility in this situation is given by $u\left(\hat{\xi}_{I,ny}, [\xi^{*,1:k_0}_{T,p}(\hat{\xi}_{I,y}); \xi^{k_0:N}_{T,brake}]\right)$.

- $u(\hat{\xi}_{I,ny}, \hat{\xi}_{T,y})$ is the utility if the target vehicle chooses to yield. For this scenario, it does not matter whether the interacting vehicle yields or not. We can directly solve for the optimal trajectory for the target vehicle with additional constraints on its trajectories. For instance, we can set an upper bound for the achievable zones of its trajectories to force it to yield [1]. Hence, $u(\hat{\xi}_{I,ny}, \hat{\xi}_{T,y}) = \min_{\xi_{T,y}} u(\hat{\xi}_{I,ny}, \xi_{T,y}(\hat{\xi}_{I,ny}))$ with constraints in the form of $g(\xi_{T,y}) \leqslant 0$.

Apart from the utilities, we also need to find an objective probability variable that can quantify approximately how the interacting vehicle will respond if the target vehicle was to take the pass action, i.e., approximating $p_{I,y}$ in (5.10) given historical observations $(\xi^t_I, \xi^t_T)$. Inspired by the predefined models, we use the TTC to approximate it. Define the TTC gap between the two interacting vehicles as $\triangle_{TTC}$=$TTC_T - TTC_I$. We assume that $p_{I,y}$ is higher if $\triangle_{TTC}$ is lower:

$$p_{I,y} = \frac{1}{1 + \exp\left(TTC_T - TTC_I\right)} \tag{5.17}$$

[1]The settings of the upper bound differ depending on scenarios. For interactions and roundabouts, the upper bound comes from the traffic-rule maps such as the locations of stop bars. For ramp merging, the upper bound can be draw from the trajectory of the interacting vehicle. We will explain more details about this in the case study.

### 5.3.3 Value Function and Decision Weighting Function Learning

With the acquired utilities and probabilities, the next step is to formulate a learning problem to find the unknown parameter $\alpha$ in the value function as well as $\gamma$ in the decision weighting function in (5.10)-(5.11). To achieve this goal, we again adopt the principle of maximum entropy to convert the decision selection process in (5.12) as a soft-max problem:

$$\mathrm{Pr}(a_p) = \frac{1}{1+\exp\left(V(a_y)-V(a_p)\right)}, \tag{5.18}$$

$$\mathrm{Pr}(a_y) = \frac{1}{1+\exp\left(V(a_p)-V(a_y)\right)}. \tag{5.19}$$

where $\mathrm{Pr}(a_p)$ and $\mathrm{Pr}(a_y)$ represent the probabilities of choosing action $a_p$ and $a_y$, respectively. Given a set of $K$ interactive trajectories with labelled decisions for the target vehicle, denoted by $S = \{(\xi_I^i, \xi_T^i, a_T^i), i{=}1, \cdots, K\}$, we can formulate the learning of $\alpha$ and $\gamma$ as a nonlinear logistic regression problem with the loss function as:

$$\begin{aligned} L(\alpha,\gamma) \quad = \quad & \sum_{i}^{K} \Big\{ -\mathbf{1}(a_T^i{=}a_p) \log \mathrm{Pr}_i(a_p) \\ & \qquad -\mathbf{1}(a_T^i{=}a_y) \log\left(1-\mathrm{Pr}_i(a_y)\right) \Big\}, \end{aligned} \tag{5.20}$$

where $\mathbf{1}(x){=}1$ if $x = 1$ and $\mathbf{1}(x){=}0$ otherwise. $\mathrm{Pr}_i(a_y)$ and $\mathrm{Pr}_i(a_p)$ are the evaluated probabilities as in (5.18)-(5.19) on the $i$-th pair of interactive trajectories $(\xi_I^i, \xi_T^i, a_T^i)$ based on (5.10)-(5.11) and (5.17).

The optimal $\alpha$ and $\gamma$ can be found via

$$(\alpha^*, \gamma^*) = \arg\min_{\alpha,\gamma} L(\alpha,\gamma). \tag{5.21}$$

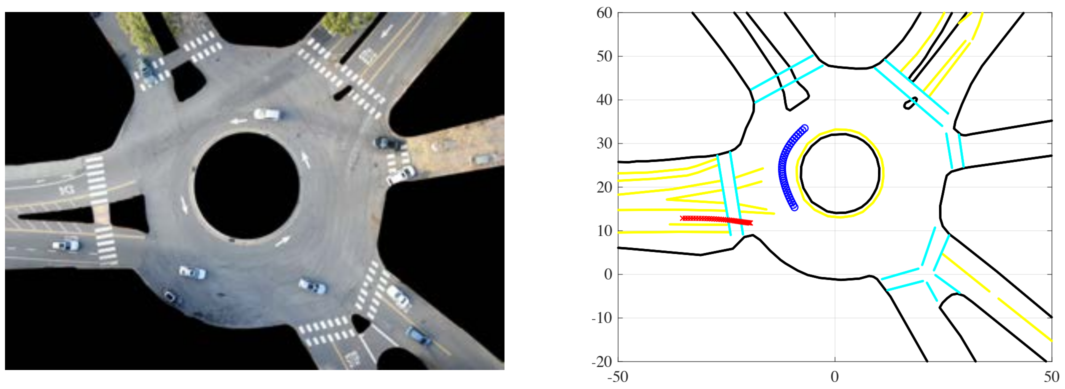
With the three steps described above, all the unknowns in the CPT model in (5.10)-(5.11) can be obtained.

## 5.4 A Case Study

### 5.4.1 A driving scenario: roundabout

To evaluate the performance of the proposed approach, we select a roundabout merging scenario from the INTERACTION dataset [158, 159]. As shown in Fig. 5.3(a), the roundabout has 6 branches and each branch has two directions (both in and out). We selected the interactive motions of two cars at the left-most branch (Fig. 5.3(b)) because there is no enforced stop signs at this branch before merging into the roundabout. This makes the interaction more intensive, and consequently creating more challenging problems.

We define the merging-in vehicle (the blue one in Fig. 5.3(b)) as the target vehicle, and the one already in the roundabout as the interacting vehicle (the red one in Fig. 5.3(b)). Based on a period of historical data on both vehicles, different driving behavior models try to predict whether the red target vehicle will decide to merge in front of the interacting vehicle in blue (i.e, the target vehicle passes), or wait to merge in until the blue car passes (i.e., the target vehicle yields).



(a) The map of the roundabout (b) One example of the interactive trajectories

**Figure 5.3:** The map of the roundabout and one example pair of interactive trajectories. Red stars: the target vehicle; Blue circles: the interacting vehicle.

We use the Frenet frame [149] to represent the trajectory coordinates of each vehicle. Reference paths of the map are shown in Fig. 5.4(a). To capture the relationships between the two cars on the longitudinal direction, we set the crossing point of the reference paths of the two interactive cars as their shared reference point. Before the crossing point, the longitudinal coordinates of both cars are negative. Once passed the crossing point, both longitudinal coordinates become positive. One example of the interactive trajectories in the defined Frenet frame can be found in Fig. 5.4(b).

## 5.4.2 Comparison models

We compared the decision prediction performance among three different models: 1) a predefined TTC rule-based model, 2) a learning-based neural network model, and 3) the proposed CPT model. A brief introduction of each model is given below.

### The TTC rule-based model

The TTC rule-based model uses the TTC as an indicator to predict which car will go first between the two interactive cars. Given the trajectories of each car in Frenet frame

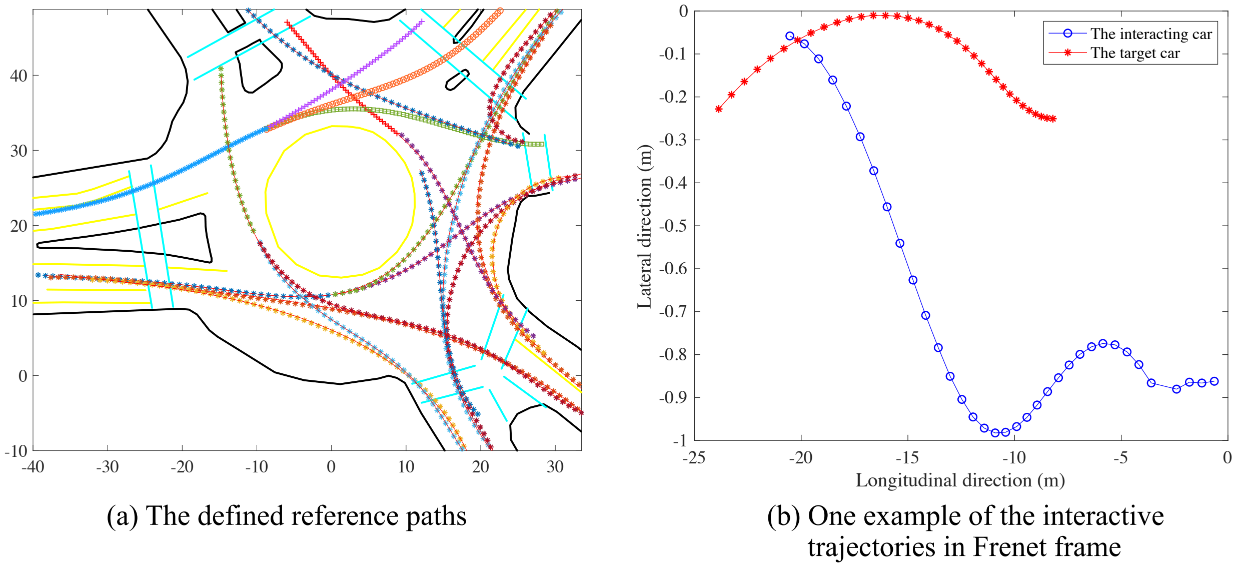


**Figure 5.4:** The reference paths (a) and trajectories in Frenet frame (b). The crossing points on the pair of reference paths define the common reference zero points for two interactive cars.

as shown in Fig. 5.4(b), the TTC can be easily calculated via

$$\mathrm{TTC}^{t}_{I,T} = s^{t}_{I,T}/v^{t}_{I,T} \tag{5.22}$$

where $s^{t}_{I,T}$ represents the longitudinal length from the current location of the cars at time $t$ to the collision point along the reference paths. $v^{t}_{I,T}$ is the current speed of the cars.

We calculate the soft-max probability of the target car passing via

$$\mathrm{Pr}^{t}(a_p) = \frac{1}{1+\exp\left(\mathrm{TTC}^{t}_{T} - \mathrm{TTC}^{t}_{I}\right)}. \tag{5.23}$$

### The learning-based neural network model

The learning-based model we used is based on neural networks (NNs). The input is a period of historical trajectories of the two interacting vehicles in Frenet frame. The first layer is a long short-term memory (LSTM) cell with 16 neurons, followed by two fully connected layers, and each with 8 neurons. Afterwards, a *tanh* nonlinear activation layer is applied, with a the softmax layer as the final layer to output the classification results. In order to avoid over-fitting, we applied the drop-out technique to the fully connected layers with a dropout rate of 0.5 and added a L2 regularization term to the original cross-entropy loss function.

### The proposed CPT model

In the proposed CPT model, we have selected four features in the utility function. They are defined as:

- Speed feature $\phi_1 = \exp\left(-(v^t - v_{\mathrm{traffic}})^2\right)$;
- Acceleration feature $\phi_2 = \exp\left(-(acc^t)^2\right)$;
- Jerk feature $\phi_3 = \exp\left(-(jerk^t)^2\right)$;
- Safety feature $\phi_4 = \exp\left((s_I^t - s_T^t)^2\right)$.

Note that all the variables $v^t$, $acc^t$ and $jerk^t$ can be written as linear functions of the trajectories of the target vehicle based on backward differentiation.

As for the calculation of key utilities in (5.10)-(5.11), examples of the corresponding trajectories for the utility evaluation are shown in Fig. 5.5. The ground truth interactive trajectories are shown in Fig. 5.5(a) with red for the target vehicle and blue for the interacting vehicle. Figure 5.5(b) shows the optimal yielding trajectory of the target vehicle (cyan) and the ground truth trajectory of the interacting vehicle (blue). Figure 5.5(c) and Fig. 5.5(d), respectively, show the trajectories of the target vehicle (cyan) under a passing decision with a non-yielding and yielding interacting vehicles. If the interacting is not-yielding, we assume that it will maintain its initial speed, as shown in green in Fig. 5.5(c). In this case, the target vehicle is forced to brake. On the other hand, with a yielding interacting vehicle, the optimal passing trajectory of the target vehicle is shown in Fig. 5.5(d).

## 5.4.3 Experiment results and discussion

We discuss the experimental results in two aspects: prediction performance comparison among the three models, and the interpretability of these models.

### Comparison of the prediction performance

We trained and tested all three models on a dataset containing 67 pairs of interacting trajectories with a sampling frequency of 10Hz. To learn more generalized results, we slice the trajectories into frames with a fixed length using moving windows. Each frame contains the trajectories in 1s. Thus, all 67 pairs of interacting trajectories generate 2680 frames. To achieve better performance for the learning-based model, we have conducted two sets of experiments for the training of the neural network:

- Experiment 1: randomly shuffle all the trajectory pairs and select 80% of them for training and the other 20% for testing. The success rate [2] is 65% for testing.

[2]Success rate is defined as the percentage of correct predictions among all test examples.

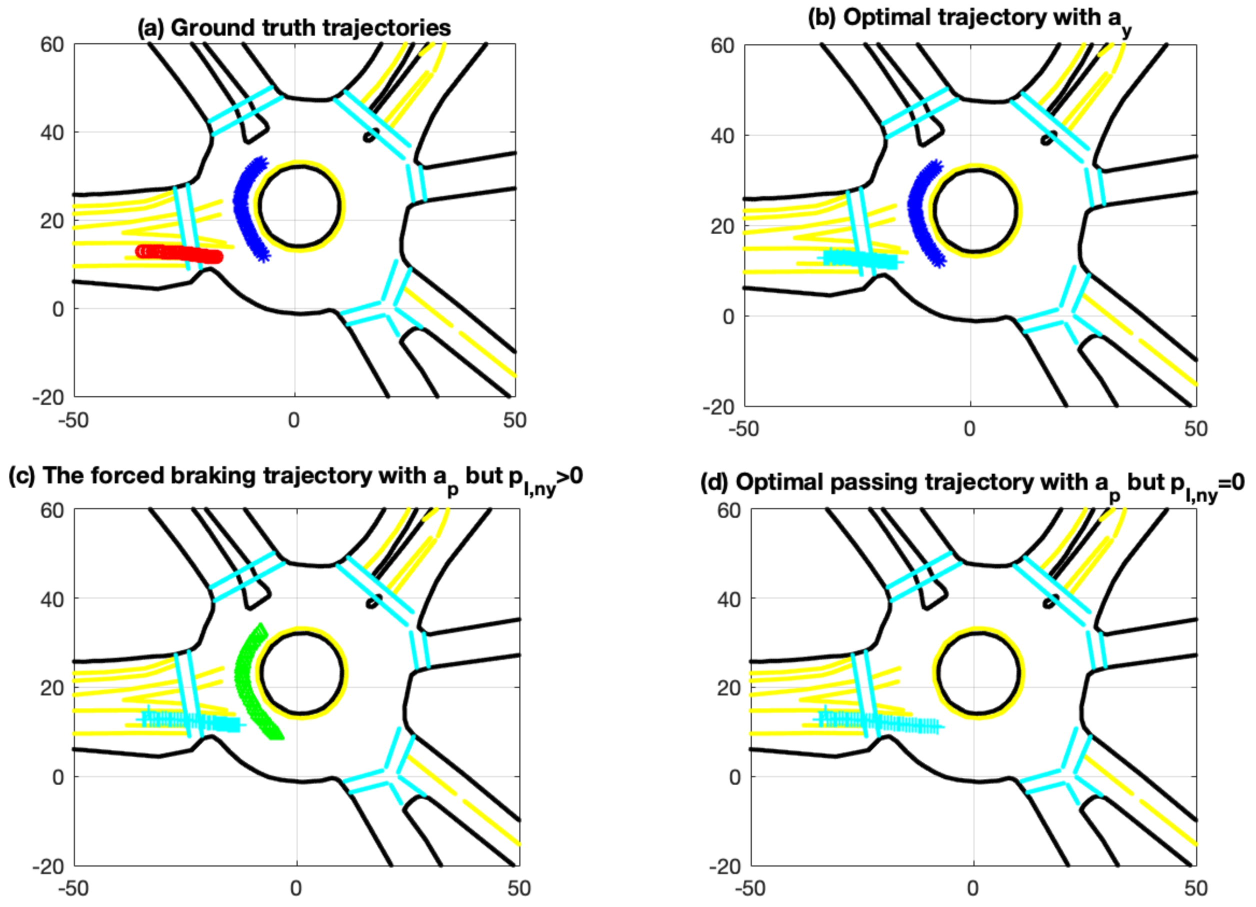


**Figure 5.5:** An example of the trajectories used for utility calculation under different decisions and different responses of the interacting vehicle: (a) the ground truth trajectories (red: the target car; blue: the interacting car); (b) the optimal trajectory of the target car (cyan) under the decision of yielding ($a_p$), and the ground truth of the interacting car (blue); (c) the forced braking trajectory of the target car (cyan) under a passing decision but with a non-yielding interacting car (green). The virtual trajectory of the interacting car is assumed to maintain its initial speed; (d) the optimal trajectory of the target car (cyan) under a passing decision with a yielding interacting car.

- Experiment 2: directly shuffle all frames for the neural network and randomly select 80% for training and 20% for testing. The success rate is 97% for testing.

The large discrepancy between the testing accuracies of the two experiments with the NN model is mainly due to the over-fitting problem cause by the data insufficiency. In experiment 1, it showed that the NN model learned on 80% of the trajectory pairs cannot be well generalized to other interaction pairs.

We list the success rates for prediction from all three models in Table 5.1. It shows that the proposed CPT model outperformed the TTC model and the NN model in experiment 1, and it achieved similar performance as the NN model in experiment 2. Moreover, both the TTC model and the proposed CPT model are more data-efficient for similar achievable performance.

**Table 5.1:** Comparison of the success rates in three models

| | TTC | NN | CPT |
|---|---|---|---|
| Success rates | 81.82% | 97% | 95.45% |

## Interpretability of the CPT model

In the CPT model, the parameters we have learned via the nonlinear logistic regression are

$$\alpha^* = 0.9827, \gamma^* = 0.6742. \tag{5.24}$$

With the optimal $\gamma$, the learned decision weighting function is shown in Fig. 5.6. We can see that the CPT model indeed captured the human choice patterns that events with low probabilities will tend to be overestimated, while high-probability events are often underestimated. Such results are consistent with many studies about human behavior in other domains such as economics, investment and waiting paradox problems.

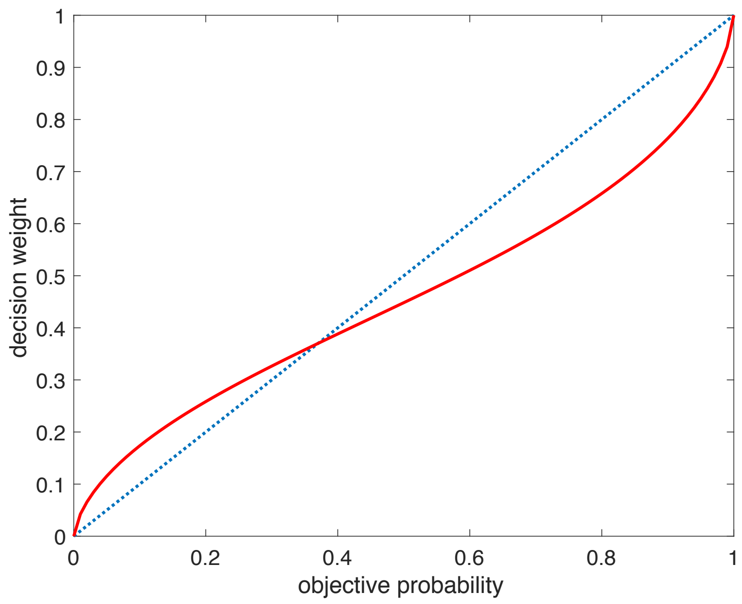


**Figure 5.6:** The learned decision weighting function (red curve) shows a S shape which is consistent with the results obtained in other domains for human behavior such as economics, investment and waiting paradox problems.

## 5.5 Chapter Summary

In this chapter, we addressed the problem of irrationality-compatible reward design and learning, using autonomous vehicles as an application example. We proposed an interpretable and irrationality-aware human behavior model based on the cumulative prospect theory (CPT). To learn the model parameters from human demonstrations, a hierarchical learning algorithm was developed, in which inverse reinforcement learning and nonlinear logistic regression were combined. We also have conducted comparison studies among three different behavior modelling approaches: a predefined TTC model, a neural network (NN) based learning model, and the proposed CPT model. The results showed that the proposed CPT model outperformed the TTC model in terms of prediction accuracy. Similar performance was achieved by the CPT model as the NN model, but with much less amount of data. Moreover, the learned parameters of the CPT model have explicit and interpretable physical meanings, which matched the observations of the human behavior in many domains.

The work in this chapter of course cannot explain all irrational behaviors of human. Reasons for irrational behaviors of human can be many. For instance, they can be caused by different perception views or incomplete perceptions, different beliefs about the behavior models of others or inaccurate/unreliable execution of the decisions. In this chapter, we try to cover the irrationality caused by a different measure during their optimizations. The work can be further extended in many directions, for instance, the inclusion of online inference for dynamic reference points. However, we are encouraged to see that by enforcing the structure of the reward design, more interpretable and irrationality-compatible behavior model can be obtained with similar performance compared to deep learning models.

Chapters 3 - 5 have addressed three different aspects of human bebavior, i.e., the social compliance, hierarchy and irrationality, and introduced them in the reward design and learning process. Applications on autonomous vehicles have been demonstrated. The results show that via appropriate structure design and parameter learning, reward functions can well represent a variety of human behavior with an interpretable model. Moreover, the learned reward functions can be directly utilized in the robot's behavior generation policies via optimization, which can help generate more human-like behaviors or improve the prediction performance of the robots.

# Chapter 6

# Behavior Planning Under Perception Uncertainties

Through Chapters 3 - 5, reward functions to present human behaviors have been designed and learned via human demonstrations. To enable safe, efficient and human-like interactive behaviors of the autonomous systems, the next question is how to generate interaction-aware behaviors in an efficient and reliable manner in the presence of uncertainties.

Uncertainties are everywhere for autonomous systems. Starting from localization, every module of the system can generate or suffer from uncertainties. For instance, we have localization uncertainties, perception uncertainties, behavior modelling uncertainties, behavior policy uncertainties, and hardware-level model uncertainties at the execution phase.

In the following two chapters, we will address the behavior planning of autonomous system in the presence of perception uncertainties and interactive policy uncertainties, using autonomous driving as an application example. In Part II, we will discuss about hardware-level model uncertainties and external disturbances with exemplar application domains in precision motion systems.

## 6.1 Introduction

The driving environment of autonomous vehicles (AVs) are dynamic and can be full of uncertainties. First, the future behaviors and trajectories of other traffic participants, such as pedestrians or vehicles with human drivers, are probabilistic in nature. It is difficult to predict them precisely, particularly in highly interactive driving scenarios. Beyond that, the implicit social behavior on local driving preferences and styles is also hard to describe exactly when the AVs are adapting themselves to a new environment. Moreover, the detection and tracking modules can produce lots of physical state uncertainties due to the algorithmic limitation in terms of unsatisfactory performance, as well as the physical limitation such as sensor field-of-view occlusions and limited sensor range.

To generate safe and efficient maneuvers of autonomous vehicles, the decision-making and planning modules of AVs should be able to properly tackle all the uncertainties in the preceding modules such as perception and prediction. Research efforts were devoted recently to designing decision-making and planning algorithms under behavioral uncertainties from prediction. For example, an interactive belief-state planner proposed in [51] used Partially Observable Markov Decision Process (POMDP) to deal with the behavior uncertainties of other vehicles. A decision-making framework was also constructed in [93] to deal with uncertain behavior of other vehicles at intersections considering potential violations.

While focusing on the behavior uncertainties, there is a common assumption in the aforementioned work, that is, the physical states of other traffic participants are deterministic and accurate. The perception module is assumed to provide perfect results on whether an object exist or not, or what the current positions, velocities and orientations of different objects are. However, such an assumption can hardly hold in practice. Even if the sensors, such as cameras and LiDARs, can well capture the objects, it is impossible for the most state-of-the-art algorithms to achieve perfect perception, as evidenced by the 3D detection results for the "easy" cases on KITTI benchmark [140].

In addition to the algorithmic limitations, the physical limitations can also lead to uncertainties of the physical states of objects. Physical limitations in perception mainly include occlusion and limited sensor range [97]. Occlusion, an inevitable encounter of autonomous vehicles, causes great challenges for tracking, prediction and risk assessment [37] [77][155], and therefore poses significant impacts on the performance of decision-making and planning. To deal with occlusions, a safe driving strategy was proposed in [44] at blind intersections. Focusing also on blind intersections, [90] directly designed a planning method based on inverse reinforcement learning (IRL). In [20], a decision-making approach under occlusions was proposed in the framework of POMDP.

Most of the works, however, treats all other traffic participants such as pedestrians or human-driven vehicles only as objects/obstacles to avoid. In fact, they are all intelligent agents whose behaviors can be quite informative. *Hence, our key observation is that human participants should be treated not only as dynamic obstacles, but also as distributed sensors that provide via their behaviors additional information about the environment beyond the scope of physical sensors. We call this concept **social perception**. The decision-making and planning modules of autonomous cars should explicitly exploit the enhancement offered by the social perception.*

Figure 6.1 demonstrates several exemplar scenarios where other road users can serve as sensors to overcome occlusions or limited sensor range. In Fig. 6.1(a), the host vehicle V0 cannot detect the pedestrian due to the occlusion caused by V1 and V2. However, it can be inferred that the most probable reason for V1 to decelerate is a pedestrian crossing the street. Therefore, the behavior of V1 can be exploited as a sensor to enable social perception for potential pedestrians. In Fig. 6.1(b), the host vehicle V0 is making a

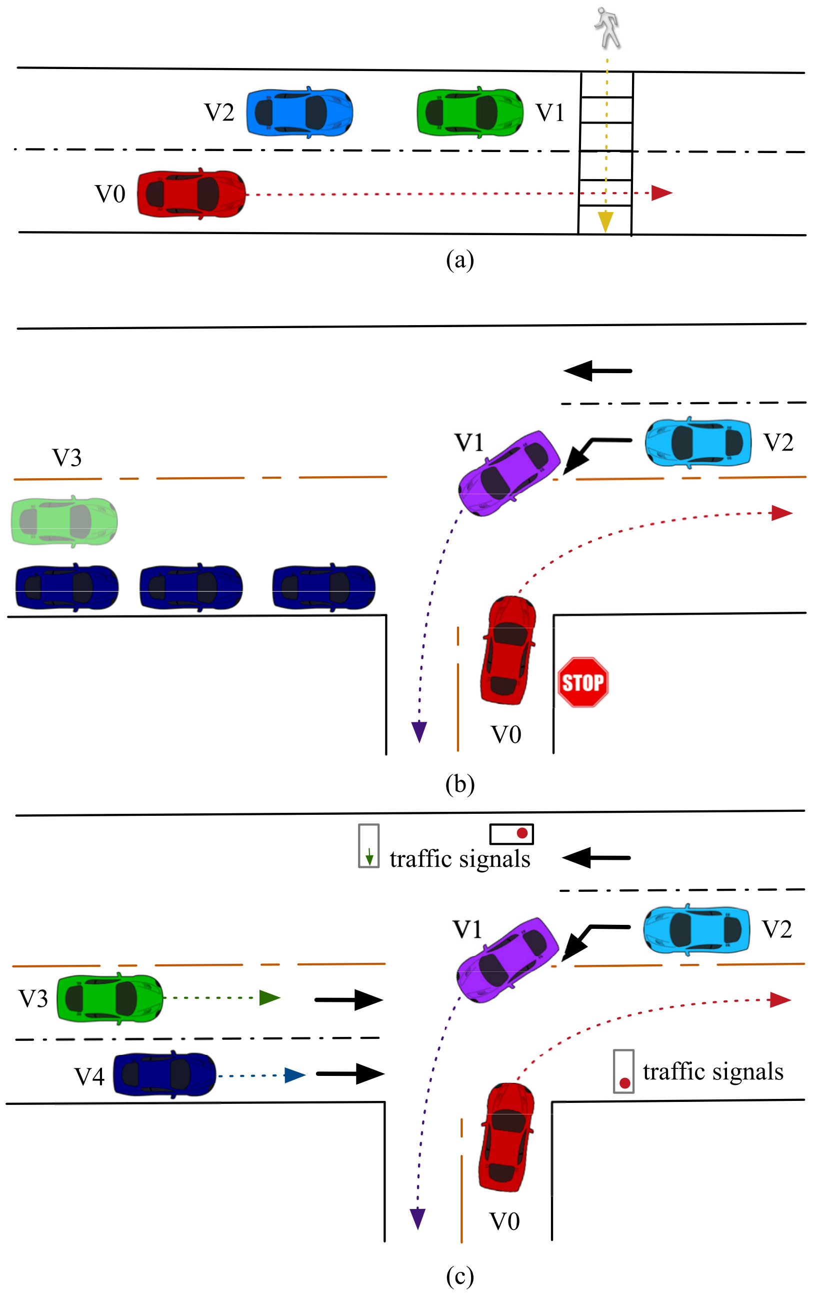


**Figure 6.1:** Exemplar scenarios where other road users can serve as distributed social sensors for the host vehicle (red): (a) although the host vehicle V0 cannot detect the pedestrian due to the occlusion caused by V1 and V2, it can infer from the behavior of V1. (b) The right-turning vehicle V0 can infer the existence of occluded vehicle V3 from the behaviors of the left-turning vehicle V1. Similarly, in (c), the right-turning vehicle at a signalized intersection can infer about the availability of the intersection by observing the actions of the left-turn-only vehicles V1 and V2.

right turn at a one-way-stop T-intersection. It should yield to a potential vehicle V3, but the view is occluded by street-parked vehicles. However, V1 and V2 on the left-turn-only lane keep moving and making left turn, which indicates that there should be no vehicle in the occluded area or a vehicle if any might be relatively far away, and V0 may proceed to turn. Figure 6.1(c) shows a signalized intersection. The host vehicle V0 (turning right) can only detect the signal (red light) in front of it controlling its direction. It should yield to V3 and V4 which are still with relatively high speeds. However, V1 and V2 on the left-turn-only lane accelerate, which indicates that there is a protected left turn for them, and V0 can proceed to turn right. Therefore, the social perception is needed when the motion attributes of others are out of the limited sensor range.

Inferring the physical states from the behavior of others as described in the examples is one aspect of social perception. For example, [4] infers the map occupancy from behaviors of human drivers based on manually designed rules. A more important aspect for social perception is that it can go beyond the perception of physical states and extend to the perception of social information existed within a group of social agents. Courtesy [135] is one of the representative social information to be extracted. Socially cohesive behavior was analyzed and designed in [67] by assuming that the behaviors of others (for instance, human drivers) were often correct and similar behaviors should be generated by autonomous vehicles.

Integrating the social perception into the decision-making and planning modules of autonomous vehicles is extremely important to enable safer and more efficient maneuvers in the presence of corresponding uncertainties. Collisions could be potentially avoided (Fig. 6.1(a)) and the behavior of the autonomous vehicle can be more efficient, less conservative (Fig. 6.1(b) and (c)), and more socially compatible so that both the passengers and the other human drivers will not be surprised or annoyed. In this chapter, we explicitly incorporate the social perception into a probabilistic planner based on Model Predictive Control (MPC), and propose a unified planning framework to handle the above mentioned uncertainties.

## 6.2 Problem Statement

In this chapter, we consider the behavior planning of an autonomous car in a multi-agent environment with perception uncertainties. Except for the autonomous car, denoted as $\mathcal{R}$, we assume all other agents to be human, represented by $\mathcal{H}$. Hence, we do not explicitly model the interactions among human, but focus on the interaction between the robot car and an individual human. As for the perception uncertainties, we consider two types of uncertainties as defined above: the physical state uncertainties such as occlusions and limited sensor range, and the social behavioral uncertainties such as local driving preferences.

Throughout the chapter, we let $x_{\mathcal{R}}$ and $u_{\mathcal{R}}$ denote, respectively, the robot car's states and control inputs, and $x_{\mathcal{H},i}$ and $u_{\mathcal{H},i}$ for those of human $i$. In a traffic scene with $M$

human participants, the states of all agents become $x_A{=}(x_{\mathcal{R}}^T, x_{\mathcal{H},1}^T, \cdots, x_{\mathcal{H},M}^T)^T$, and the environment states can be represented by $x{=}(x_A^T, x_{Env}^T)^T$ where $x_{Env}$ is the non-agent related states such as traffic lights. We use $\mathcal{I}$ to represent the social information set. For each agent, we have

$$\begin{aligned} x_{\mathcal{R}}^{t+1} &= f_{\mathcal{R}}\left(x_{\mathcal{R}}^t, u_{\mathcal{R}}^t\right), && (6.1)\\ x_{\mathcal{H},i}^{t+1} &= f_{\mathcal{H},i}\left(x_{\mathcal{H},i}^t, u_{\mathcal{H},i}^t\right), i{=}1, \cdots, M, && (6.2)\end{aligned}$$

where $f_{\mathcal{R}}$ and $f_{\mathcal{H},i}$ describe, respectively, the dynamics of the autonomous car and human $i$. The closed-loop dynamics of the whole multi-agent system becomes

$$x^{t+1} = f(x^t, u_{\mathcal{R}}^t, u_{\mathcal{H},1}^t, \cdots, u_{\mathcal{H},M}^t). \tag{6.3}$$

We assume that all agents in the scene are noisily optimal planners. Namely, at time $t$, each agent behaves to minimize its own cost function based on its estimates of the environment states ($\hat{x}^t$) and the social information ($\hat{\mathcal{I}}^t$) inferred from observations, denoted by $o^t$. Let $C_{\mathcal{R}}$ and $C_{\mathcal{H},i}$ be, respectively, the cost functions of the robot car and human $i$ at time $t$ over a horizon of $N$:

$$C_j\left(\hat{x}^t, \hat{\mathcal{I}}^t, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H},i}; \theta_j\right) = \sum_{k=0}^{N-1} c_j\left(\hat{x}^t, \hat{\mathcal{I}}^t, u_{\mathcal{R}}^k, u_{\mathcal{H},i}^k; \theta_j\right), j{\in}\{\mathcal{R}, (\mathcal{H}, i)\}$$

where $\mathbf{u}_j{=}(u_j^0, u_j^1, \cdots, u_j^{N-1})^T$ is the sequence of control actions of the agent within the horizon ($j{=}\mathcal{R}$ for the robot car and $j{=}(\mathcal{H}, i)$ for human $i$). $\theta_j$ with $j{\in}\{\mathcal{R}, (\mathcal{H}, i)\}$ represent, respectively, the preferences of the robot car and human $i$. At every time step $t$, all agents generate their optimal sequences of actions $\mathbf{u}_{\mathcal{R}/\mathcal{H},i}^*$ by minimizing their corresponding cost functions $C_{\mathcal{R}/\mathcal{H},i}$, execute the first steps $u_{\mathcal{R}/\mathcal{H},i}^{*0}$ (i.e., set $u_{\mathcal{R}/\mathcal{H},i}^t{=}u_{\mathcal{R}/\mathcal{H},i}^{*0}$ in (6.1) and (6.2) and re-plan at the next time step at $t{+}1$.

As shown in (6.4), robot cars generate behaviors based on their estimates of the environment states and the social information set. If the estimates significantly deviate from the ground truth, unexpected or even dangerous behaviors might be generated. For environment states, current practice is to set the estimates as the de-noised observations of the robot car from physical sensors, i.e., $\hat{x}^t{\approx}o_{\mathcal{R}}^t$. However, due to occlusions and limited sensor ranges, the states observation $o_{\mathcal{R}}^t$ of the robot car might be a subset or even different from the actual states $x^t$, which makes $\hat{x}^t{\approx}o_{\mathcal{R}}^t$ not an effective solution. As for the social information $\mathcal{I}^t$, it is a set of variables that cannot be directly perceived by physical sensors. Hence, to enable better autonomous driving strategies under such perception uncertainties, more advanced perception/inference scheme is desired to update $\hat{x}^t$ and $\hat{\mathcal{I}}^t$ from observations.

## 6.3 Social Perception

Our key observation is that human traffic participants should be treated not only as dynamic obstacles that the robot cars need to be aware of, but also as distributed sensors the behaviors of which can provide additional information beyond the scope of physical sensors equipped with the autonomous vehicles.

### 6.3.1 Distributed Agents as Distributed Sensors

Again consider the multi-agent system consisting of one robot car and M humans. With physical sensors subjected to occlusions and limit range, each agent can observe only a subset of the environment states, denoted by $o_j^t$ with $j \in \{\mathcal{R}, (\mathcal{H}, i)\}$. Based on the corresponding observation $o_j^t$, each agent extracts their estimates, $\hat{x}_j^t$ and $\hat{\mathcal{I}}_j^t$, on the environment states and the social information, respectively. The estimates of different agents will then influence their next-step actions/trajectories which can be perceived by other agents. Note that due to their distributed locations, observations and the associated estimates of different agents can significantly differ, but be complementary to each other. Hence, the distributed agents can be viewed as distributed sensors which emit behavioral signals. By observing such behavioral signals, the robot car can infer estimates $\hat{x}_j^t$ and $\hat{\mathcal{I}}_j^t$ from human to reduce its perception uncertainties coming from either algorithmic limitations, or physical limitations, or both.

### 6.3.2 Inference Algorithm

**Modeling the behavior generation function of human**

As discussed in Section 6.2, we assume that each human is a noisily optimal planner, and consider the interaction between the robot car and humans when modeling the humans' behaviors. Thus, at each time period (N steps) starting at t, the behavior sequence $\mathbf{u}_{\mathcal{H}}^t = [u_{\mathcal{H}}^t, u_{\mathcal{H}}^{t+1}, \cdots, u_{\mathcal{H}}^{t+N-1}]^T$ minimizes the human's cost function as given in (6.4) based on his/her estimates. Namely, the behavior generation function of the human can be expressed as

$$\mathbf{u}_{\mathcal{H}}^{*,t} = \arg\min_{\mathbf{u}_{\mathcal{H}}} C_{\mathcal{H}}\left(\hat{x}_{\mathcal{H}}^t, \hat{\mathcal{I}}_{\mathcal{H}}^t, \mathbf{u}_{\mathcal{R}}^t, \mathbf{u}_{\mathcal{H}}; \theta_{\mathcal{H}}\right) \triangleq g_{\mathcal{H}}\left(\hat{x}_{\mathcal{H}}^t, \hat{\mathcal{I}}_{\mathcal{H}}^t, \mathbf{u}_{\mathcal{R}}^t; \theta_{\mathcal{H}}\right), \tag{6.4}$$

and the optimal cost is given by

$$C_{\mathcal{H}}^{*,t}(\mathbf{u}_{\mathcal{R}}^t) = C_{\mathcal{H}}\left(\hat{x}_{\mathcal{H}}^t, \hat{\mathcal{I}}_{\mathcal{H}}^t, \mathbf{u}_{\mathcal{R}}^t, \mathbf{u}_{\mathcal{H}}^{*,t}; \theta_{\mathcal{H}}\right). \tag{6.5}$$

Note that in (6.4) - (6.5), we model the human behavior generator $g_{\mathcal{H}}$ as his/her optimal response function to the robot car's input $\mathbf{u}_{\mathcal{R}}^t$ to explicitly address the influences from the robot car, as in [118]. Hence, if the robot car can access the humans' cost function parameters $\theta_{\mathcal{H}}$ and their estimates, it can calculate the best behavioral responses $\mathbf{u}_{\mathcal{H}}^{*,t}$ from them.

**Updating beliefs on estimates via inference**

To use humans as sensors of the environment, we need to construct observation models for the robot car to update its beliefs on estimates. For environment states and social information, different observation models are designed.

*Updating beliefs on state estimates.* At every step $t$, the robot can update its beliefs on the state estimates $\hat{x}^t$ from behaviors of human $i$ via:

$$b_{\mathcal{R}}(\hat{x}^t | u_{\mathcal{H},i}^t) \propto b_{\mathcal{R}}(\hat{x}^t) P(u_{\mathcal{H},i}^t | \hat{x}^t). \tag{6.6}$$

$P(u_{\mathcal{H},i}^t | \hat{x}^t)$ is the probability/likelihood of human $i$ taking action $u_{\mathcal{H},i}^t$ if the human's estimates were indeed $\hat{x}^t$. To get $P(u_{\mathcal{H},i}^t | \hat{x}^t)$, we assume that actions with higher cost are exponentially less likely based on maximum entropy principle as in [166]. This means that $P(u_{\mathcal{H},i}^t | \hat{x}^t)$ can be approximated by:

$$P(u_{\mathcal{H},i}^t | \hat{x}^t) \propto e^{-Q^*(\hat{x}^t, \hat{\mathcal{I}}^t, \mathbf{u}_{\mathcal{R}}^t, u_{\mathcal{H},i}^t; \theta_{\mathcal{H},i})} \tag{6.7}$$

where $Q^*(\hat{x}^t, \hat{\mathcal{I}}^t, \mathbf{u}_{\mathcal{R}}^t, u_{\mathcal{H},i}^t; \theta_{\mathcal{R}})$ represents the optimal cost to go by taking action $u_{\mathcal{H},i}^t$, given by

$$Q^*(\hat{x}^t, \hat{\mathcal{I}}_i^t, \mathbf{u}_{\mathcal{R}}^t, u_{\mathcal{H},i}^t; \theta_{\mathcal{H},i}) = \min_{\mathbf{u}_{\mathcal{H},i}^{1:N-1}} \sum_{k=1}^{N-1} c_{\mathcal{H}}\left(\hat{x}^t, \hat{\mathcal{I}}^t, u_{\mathcal{R}}^k, u_{\mathcal{H},i}^k; \theta_{\mathcal{H},i}\right). \tag{6.8}$$

*Updating beliefs on social information.* As defined in Section 9.1, social information refers to the group behaviors of human in the traffic scene. Therefore, to update the beliefs on estimates of social information $\hat{\mathcal{I}}$, the robot car need to collect the common behaviors from multiple human. Therefore, the belief update process becomes:

$$b_{\mathcal{R}}(\hat{\mathcal{I}}^t | u_{\mathcal{H}}^t) \propto b_{\mathcal{R}}(\hat{\mathcal{I}}^t) \prod_{i=1}^{M} P(u_{\mathcal{H},i}^t | \hat{\mathcal{I}}^t), \tag{6.9}$$

where $P(u_{\mathcal{H},i}^t | \hat{\mathcal{I}}^t)$ is the probability/likelihood of human $i$ taking action $u_{\mathcal{H},i}$ if the human's estimates on social information is indeed $\hat{\mathcal{I}}^t$. It can be evaluated in a similar way as (6.7) and (6.8) via:

$$P(u_{\mathcal{H},i}^t | \hat{\mathcal{I}}^t) \propto e^{-Q^*(\hat{x}_t, \hat{\mathcal{I}}^t, \mathbf{u}_{\mathcal{R}}^t, u_{\mathcal{H}}^t; \theta_{\mathcal{H},i})}. \tag{6.10}$$

### 6.3.3 Learning cost functions of human

As discussed above, for the robot car to update its beliefs by collecting behavioral information from of human, the robot car needs to have access to the cost functions of human, so that it can evaluate $P(u^t_{\mathcal{H}}|\hat{x}^t)$ and $P(u^t_{\mathcal{H}}|\hat{\mathcal{I}}^t)$. One can obtain such cost functions via inverse reinforcement learning (IRL) [1, 166, 73, 132]. Note that during the learning process, we assume that the demonstrations are sub-optimal and there is no perception uncertainties, i.e., $\hat{x}^t{=}x^t$. A brief review of the IRL algorithm is given below.

The single-step cost is assumed to be parametrized as a linear combination of features (the social information $\mathcal{I}$ is assumed to be invariant within one horizon):

$$c(x^t, \mathcal{I}, u^t_{\mathcal{R}}, u^t_{\mathcal{H}}; \theta) = \theta^T \phi(x^t, \mathcal{I}, u^t_{\mathcal{R}}, u^t_{\mathcal{H}}). \tag{6.11}$$

Over a horizon of N, the cumulative cost function is

$$C(x^0, \mathcal{I}, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}}; \theta) = \theta^T \sum_{t=0}^{N-1} \phi(x^t, \mathcal{I}, u^t_{\mathcal{R}}, u^t_{\mathcal{H}}). \tag{6.12}$$

Our goal is to find the weights $\theta$ which maximizes the likelihood of the demonstration set $\mathcal{U}_D$:

$$\theta^* = \arg\max_{\theta} P(\mathcal{U}_D|\theta) \tag{6.13}$$

Building on the principle of maximum entropy, we assume that trajectories are exponentially more likely when they have lower cost:

$$P(\mathbf{u}_{\mathcal{H}}, \theta) \propto \exp\left(-C(x^0, \mathcal{I}, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}}; \theta)\right). \tag{6.14}$$

Thus the probability (likelihood) of the demonstration set becomes

$$P(\mathcal{U}_D|\theta) = \Pi_{i=1}^n \frac{P(\mathbf{u}^D_{\mathcal{H},i}, \theta)}{P(\theta)} = \Pi_{i=1}^n \frac{P(\mathbf{u}^D_{\mathcal{H},i}, \theta)}{\int P(\tilde{\mathbf{u}}_{\mathcal{H}}, \theta) d\tilde{\mathbf{u}}_{\mathcal{H}}} \tag{6.15}$$

where $n$ is the number of trajectories in $\mathcal{U}_D$. For more details, one can refer [74] or Section 2.5 in Chapter 2 as references.

## 6.4 Human-Like Behavior Planning with Social Perception

In this section, we will discuss how to integrate the social perception into the decision-making and planning module to enable a more human-like driving strategy in terms of defensiveness, non-conservativeness, and social compatibility.

### 6.4.1 The behavior planner under uncertainties

Due to the probabilistic nature of beliefs in (6.6) and (6.9), we utilize a probabilistic framework based on Model Predictive Control (MPC) as in [157] as the planner for the autonomous cars. The cost function of the robot car is defined as an expected cost over the beliefs:

$$C_{\mathcal{R}} = \mathbb{E}_{b(\hat{x}^t),b(\hat{\mathcal{I}}^t)} C_{\mathcal{R}}\left(\hat{x}^t, \hat{\mathcal{I}}^t, \mathbf{u}_{\mathcal{R}}; \theta_{\mathcal{R}}\right) \mathbb{E}_{b(\hat{x}^t),b(\hat{\mathcal{I}}^t)} \sum_{k=0}^{N-1} c_{\mathcal{R}}\left(\hat{x}^{t,k}, \hat{\mathcal{I}}^t, u_{\mathcal{R}}^k; \theta_{\mathcal{R}}\right)$$

where $C_{\mathcal{R}}\left(\hat{x}^t, \hat{\mathcal{I}}^t, \mathbf{u}_{\mathcal{R}}; \theta_{\mathcal{R}}\right)$ is a cumulative cost over a horizon of $N$, as defined in (6.4). Note that with a long horizon $N$, discrete representation of $\hat{x}^t$ and $b(\hat{x}^t)$ is practically not feasible. In this case, we will use representative motion patterns to represent $\hat{x}^t$ as in [161].

*Cost function design.* We consider safety, efficiency, comfort, and fuel consumption in the cost. Thus, we penalize the following terms and the weight of each term can be learned via IRL as addressed in Section III-C.

- tracking error: $c_{\mathcal{R},err}=d(x_{\mathcal{R}}^{t,k})$ where $d(x_{\mathcal{R}}^{t,k})$ is the distance from the position of the robot car at $k$ time step to the desirable traffic-free reference path.
- safety term: we use the relative distances from surrounding participants to evaluate the safety term. Define $c_{\mathcal{R},safe}=\sum_{i=1}^{M} e^{-d_i(x_{\mathcal{R}}^{t,k}, x_{\mathcal{H},i}^{t,k})}$, where $M$ is the number of surrounding cars and $d_i$ is the distance of the robot car to the $i$-th one. Note that $x_{\mathcal{H},i}^{t,k}$ can be obtained via the behavior generator in (6.4) and the dynamics equation in (6.2) based on current beliefs $b(x^t), b(\mathcal{I}^t)$.
- efficiency: we penalize the difference between the speed of the robot car $v_{\mathcal{R}}\in x_{\mathcal{R}}$ and the traffic limit $v_{traffic}$, given by $c_{\mathcal{R},speed}=(v_{\mathcal{R}}^{t,k}-v_{traffic})^2$.
- acceleration: $c_{\mathcal{R},acc}=a_{\mathcal{R}}^{t,k}$ where $a_{\mathcal{R}}^{t,k}\in u_{\mathcal{R}}^{t,k}$ is the acceleration input at $k$ time step of the robot car.
- jerk: $c_{\mathcal{R},jerk}=a_{\mathcal{R}}^{t,k} - a_{\mathcal{R}}^{t,k-1}$.

Note that $v_{traffic}$ in the cost function belongs to the set of social information. We use $v_{traffic}$ instead of $v_{lim}$ to allow the robot car to infer current traffic speed and follow it. To assure that the robot car does not break the traffic rules, we expose the maximum allowable speed limit as a constraint below.

*Constraints.* To guarantee the feasibility of the planned trajectories, we introduce the following constraints:

- kinematics constraints: we use Bicycle model [113] to describe the kinematics model of the robot cars.

- dynamics constraints: we constrain curvatures and accelerations of the vehicle as follows:

$$|\kappa_{\mathcal{R}}^{t,k}| \leqslant \kappa_{\max}, \quad |a_{\mathcal{R}}^{t,k}| \leqslant a_{\max}, k{=}0, 1, \cdots, N{-}1. \tag{6.16}$$

  where $\kappa^{t,k}$ is the curvature of the planned trajectory at $k$-th step, and $\kappa_{\max}$ is the boundary of feasible curvatures. Both $\kappa_{\max}$ and $a_{\max}$ can be calculated via the "G-G" diagram as in [113].

- safety constraints: safety constraints come from both static road structures as well as dynamic obstacles such as human drivers and pedestrians. For static structures, we use polygons to represent them, and check the robot car's distance to the polygons. For dynamic obstacles, we use several circles to cover them, and calculate distances between the robot car and the circles as in [167].

Note that in the probabilistic planner, constraints over all the beliefs should all be considered. To deal with the tailing effect, we set a threshold $\epsilon$ in practice [157]. This means that if the belief of a certain state or a certain social variable is lower than $\epsilon$, we will set the probability to zero in the expected cost function, and ignore the related constraints.

### 6.4.2 The planning framework with social perception

With the probabilistic planner in Section 6.4.1, implementation of the behavior planning framework with social perception is summarized below in Algorithm 1.

## 6.5 Simulation Results

In this section, we give an exemplar scenario with sensor occlusions to verify the effectiveness of the proposed planning framework with social perception.

Despite progresses in advanced perception and tracking algorithms, sensor occlusions are inevitable for autonomous vehicles. Consider the scenario described in Fig. 6.2, where the autonomous car (red) and a human-driven car (yellow) are driving side-by-side, and a pedestrian is about to cross the street. Due to the relative positions between the robot car and the human car, the view of the robot car is blocked by the human-driven car so that it cannot detect the pedestrian.

In such a scenario, a conservative autonomous car will assume that there might be potential out-of-view pedestrians crossing the street. Hence, it slows down to prepare for stops or to leave a larger gap with the human driver to get better view. Both strategies will sacrifice the efficiency of the autonomous car. On the other hand, an aggressive autonomous car might directly ignore the possibility of pedestrian crossing the street and plan to drive through the intersection directly, which might lead to a collision. We note that in either case, the autonomous car perceives the environment states only via its

```
Input: uncertain state vector $\hat{\mathbf{x}}_u^{t_0} \triangleq \{\hat{x}_{u,1}^{t_0}, \cdots, \hat{x}_{u,L}^{t_0}\}$ (L is the number of possible states),
    observation $o^{t_0}$, prior belief $b(\hat{\mathbf{x}}_u^{t_0})$ and $b(\hat{\mathcal{I}}^{t_0})$, human behaviors $u_{\mathcal{H},i}^{t_0}$
Output: $\hat{x}^t$, $\hat{\mathcal{I}}^t$, and $u_{\mathcal{R}}^t$ for $t > t_0$
    Initialisation: $\hat{\mathbf{x}}^{t_0} = \hat{\mathbf{x}}_{\mathcal{H}}^{t_0} = o^{t_0}$,
 1: for $t = t_0, t_0+1, t_0+2, \cdots$ do
 2:   if update posterior beliefs on uncertain states $x_u^t$, then
 3:     1) select human i based on locations,
 4:     2) $b(\hat{x}_{u,l}^{t+1}|u_{\mathcal{H},i}^t) \propto b(\hat{x}_{u,l}^t)P(u_{\mathcal{H},i}^t|\hat{x}_{u,l}^t), l=1,\cdots,L$
 5:     3) normalize $b(\hat{x}_{u,l}^t)$.
 6:     4) fuse new estimates of $\hat{x}_u^t$ via updated beliefs with other states to get $\hat{x}^t$;
 7:   end if
 8:   if update posterior beliefs on social information $\mathcal{I}^t$, then
 9:     1) $b(\hat{\mathcal{I}}^{t+1}|u_{\mathcal{H},i}^t) \propto b(\hat{\mathcal{I}}^t)\prod_{i=1}^M P(u_{\mathcal{H},i}^t|\hat{\mathcal{I}}^t)$
10:     2) normalize $b_{\mathcal{R}}(\hat{\mathcal{I}}^t)$ and update its estimate.
11:   end if
12:   behavior generation via MPC:
          substitute $\hat{x}^t$, $\hat{\mathcal{I}}^t$ and the behavior generators for human in (6.4) into the
        probabilistic MPC planner in Section 6.4.1, and solve for the optimal actions $\mathbf{u}_{\mathcal{R}}^t$.
13:   execute the first action in $\mathbf{u}_{\mathcal{R}}^t$ as $u_{\mathcal{R}}^t$.
14:   update prior beliefs: $b(\hat{x}_{u,l}^{t+1}) = b(\hat{x}_{u,l}^{t+1}|u_{\mathcal{H},i}^t)$ and $b(\hat{\mathcal{I}}^{t+1}) = b(\hat{\mathcal{I}}^{t+1}|u_{\mathcal{H},i}^t)$.
15: end for
```

**Algorithm 1:** Behavior Planning with Social Perception

own physical sensors, but completely ignores the information emitted via the behaviors of the human driver.

As shown in Fig. 6.2, the view of the human driver in this scenario is not occluded about pedestrians crossing the street, and he/she is closer to the pedestrians if there was any. Hence, from the behavior of the human driver, the robot car can actually infer and become more confident about the probability of a pedestrian crossing.

We simulated this traffic scenario with a conservative planner, an aggressive planner and our proposed planner with social perception. The sampling period for each time step is 0.1s. Both cases with and without crossing pedestrians are considered, and the results are shown in Figs. 6.3 through 6.6.

### 6.5.1 With crossing pedestrians in the occluded area

Figures 6.3 and 6.4 show the comparison results with an aggressive planner and the proposed planner when there is a crossing pedestrian in the occluded area. We can see that in Fig. 6.4, with the proposed planner, when the human driver slowed

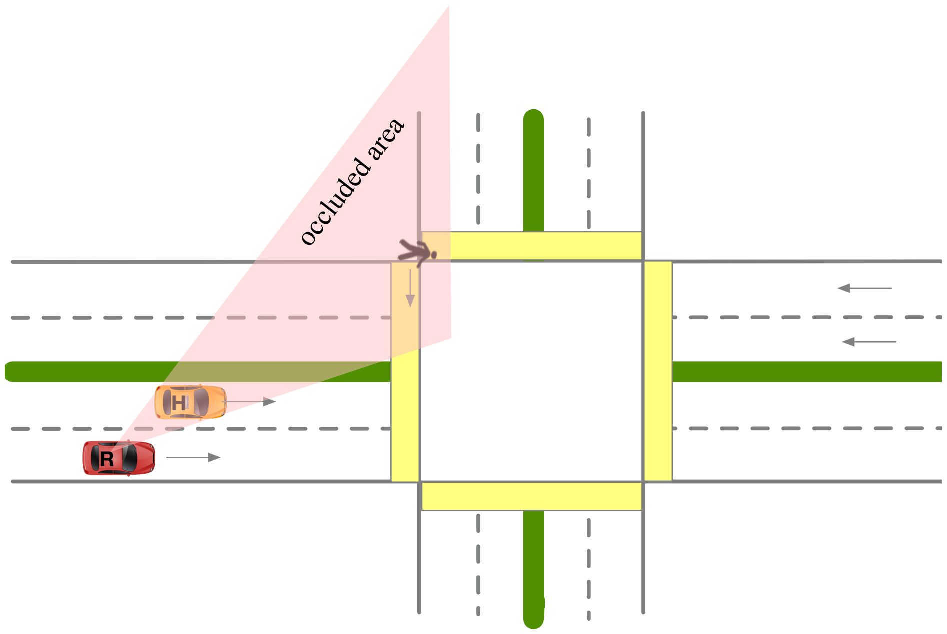


**Figure 6.2:** An example scenario of the robot car with sensor occlusions: the red robot car cannot see the pedestrian due to the occlusion caused by the yellow human car.

down, the robot car's belief on the existence of pedestrians increased quickly (Fig. 6.4(c)). Compared to the aggressive planner in Fig. 6.3, the updated belief enables the robot car to prepare for stops before occlusions are clear, while the aggressive planner failed to yield to the pedestrian even if it braked hard when it saw the pedestrian, as in Fig. 6.3.

### 6.5.2 With non-crossing pedestrians in the occluded area

We also compared the proposed planner with a conservative planner which assumes the existence of crossing pedestrians in default. Results are given in Figures 6.5 and 6.6. We can see that the proposed planner (Fig. 6.6) is much more efficient than the conservative planner (Fig. 6.5). The autonomous car with the conservative planner slowed down even if the human driver did not. On the other hand, with the proposed planner, the belief on the existence of crossing pedestrians remained low by observing the behavior of the human driver, as shown in Fig. 6.6(c), which enables the robot car to maintain relatively high speed and improves its efficiency.

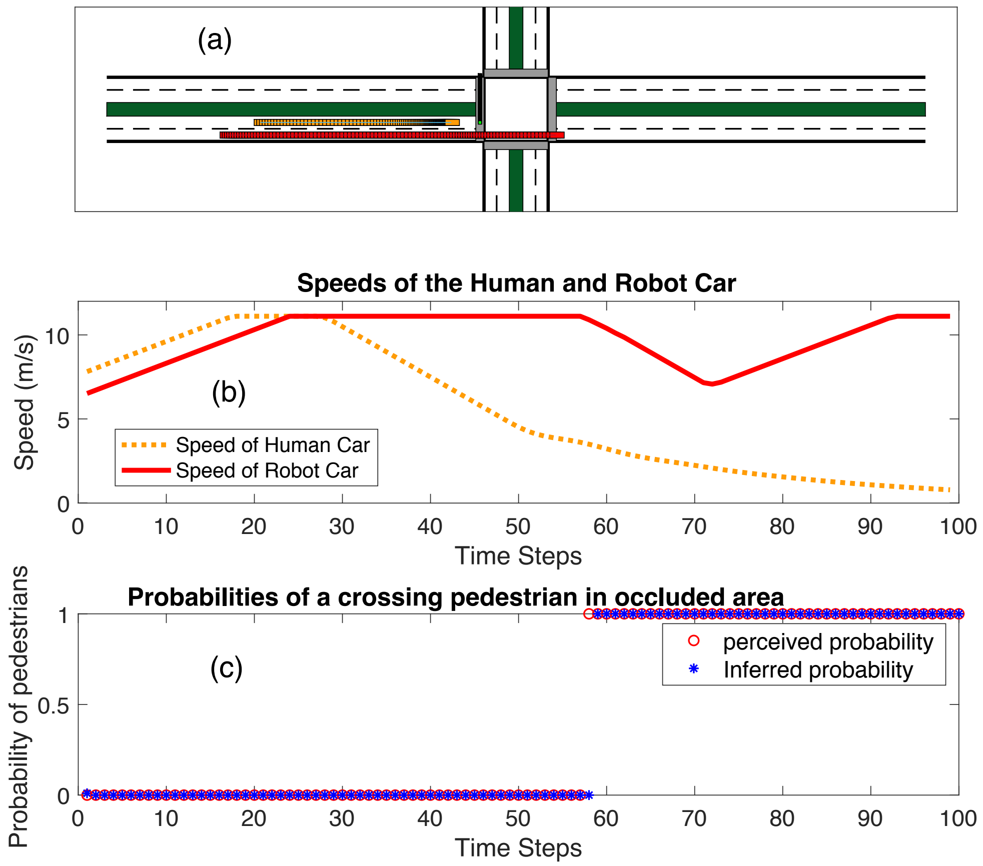


**Figure 6.3:** Simulation results with an aggressive planner with a crossing pedestrian in occluded area: (a) yellow box - human driver; red box - robot car; green dot - pedestrian; (b) speeds of the robot and human cars; (c) the perceived and inferred probabilities of pedestrians (no inference in this case)

## 6.6 Experiment Results

### 6.6.1 Experiment setup

We also have evaluated the performance on a TurtleBot3 in a similar experiment setting as the simulation. TurtleBot3 is a ROS standard platform robot developed by ROBOTIS. The MPC controller is running in MATLAB on a separate laptop with an 2.8GHz Intel Core i7-7700HQ.

As shown in Fig. 6.7, in the experiment, we let the TurtleBot3 serve as a robot car driving straight, and introduce two virtual objects in rViz: one as a pedestrian (the small green box) trying to cross the street, and the other as a large vehicle (the large green box) which is driving on the adjacent lane of the robot car and blocking part of its view.

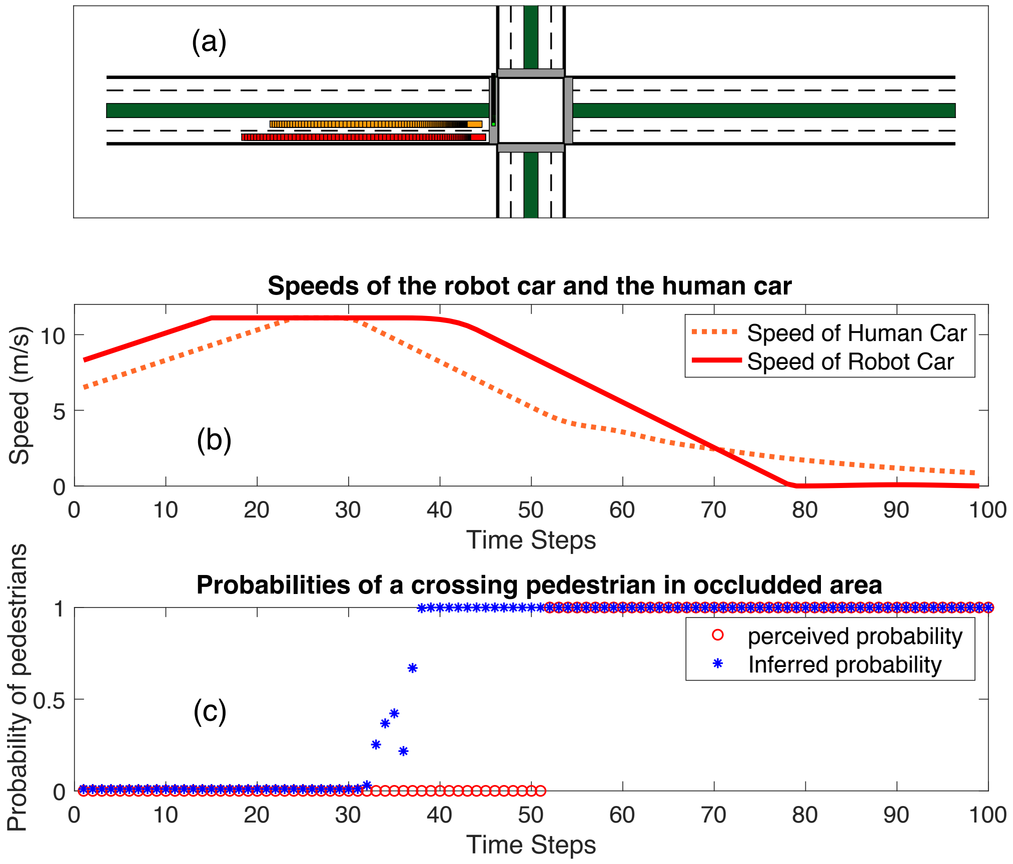


**Figure 6.4:** Simulation results with the proposed framework with a crossing pedestrian in occluded area: (a) yellow box - human driver; red box - robot car; green dot - pedestrian; (b) speeds of the robot and human cars; (c) the perceived and inferred probabilities of pedestrians

### 6.6.2 Experiment results

We switched on and off the social perception module on the TurtleBot3 to evaluate the performance of the proposed algorithm. The results are shown in Fig. 6.8. Through the sequence of screenshots, we can see that with the social perception module, the TurtleBot3 can infer in advance the existence of the virtual pedestrian and brake earlier, although its view is blocked by the large vehicle. On the contrary, without the social perception module, the TurtleBot3 did not stop until it saw the pedestrian, which might shock the pedestrian and lose trust from the pedestrian.

## 6.7 Chapter Summary

In this chapter, we proposed a unified probabilistic planning framework with social perception to deal with uncertainties from physical states, prediction of others, and un-

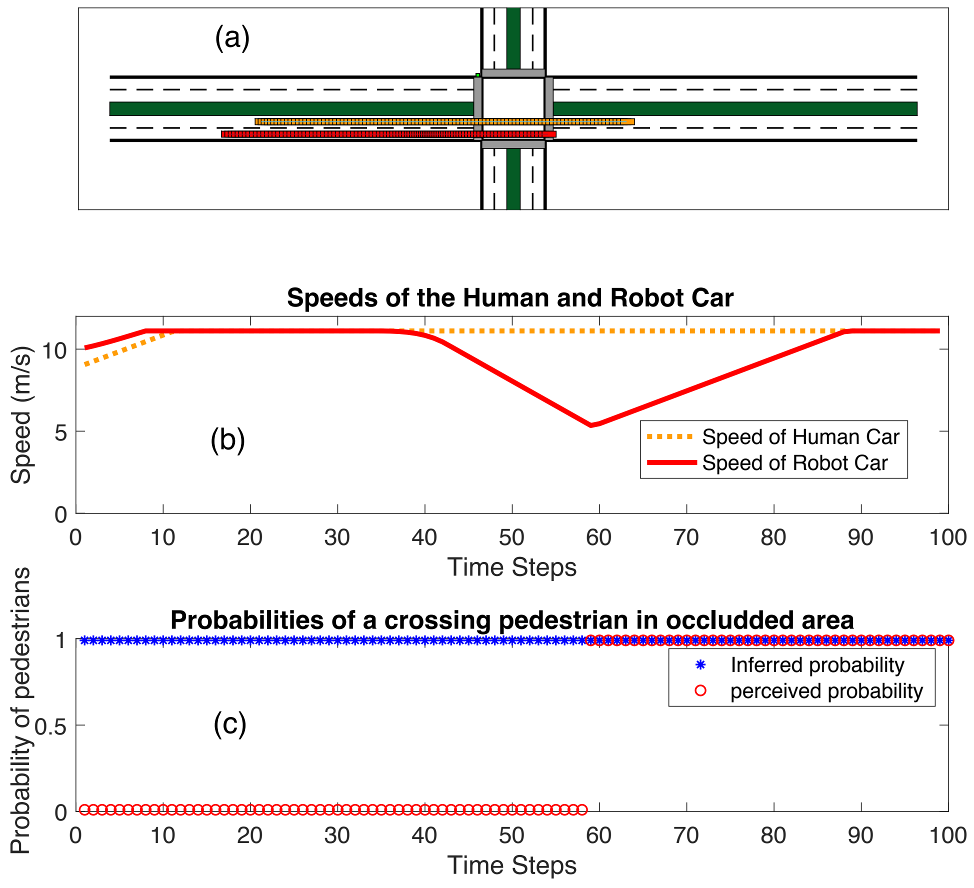


**Figure 6.5:** Simulation results with a conservative planner with non-crossing pedestrians in the occluded area: (a) yellow box - human driver; red box - robot car; green dot - pedestrian; (b) speeds of the robot and human cars; (c) the perceived and inferred probabilities of pedestrians (no inference in this case)

known social information. We treated all road participants as sensors in a distributed sensor network. By observing their individual behaviors as well as group behaviors, uncertainties of different types can be reduced via a uniform belief update process. We also explicitly incorporated the social perception scheme with a probabilistic planner based on MPC, which can thus generate behaviors which are defensive but not overly conservative, and socially compatible for autonomous vehicles. Simulation and experiment results on a turtleBot in a traffic scene with sensor occlusions were given, with comparison to a conservative planner and an aggressive planner. The results showed that the proposed framework can enable more efficient and yet defensive behaviors in the presence of perception uncertainties.

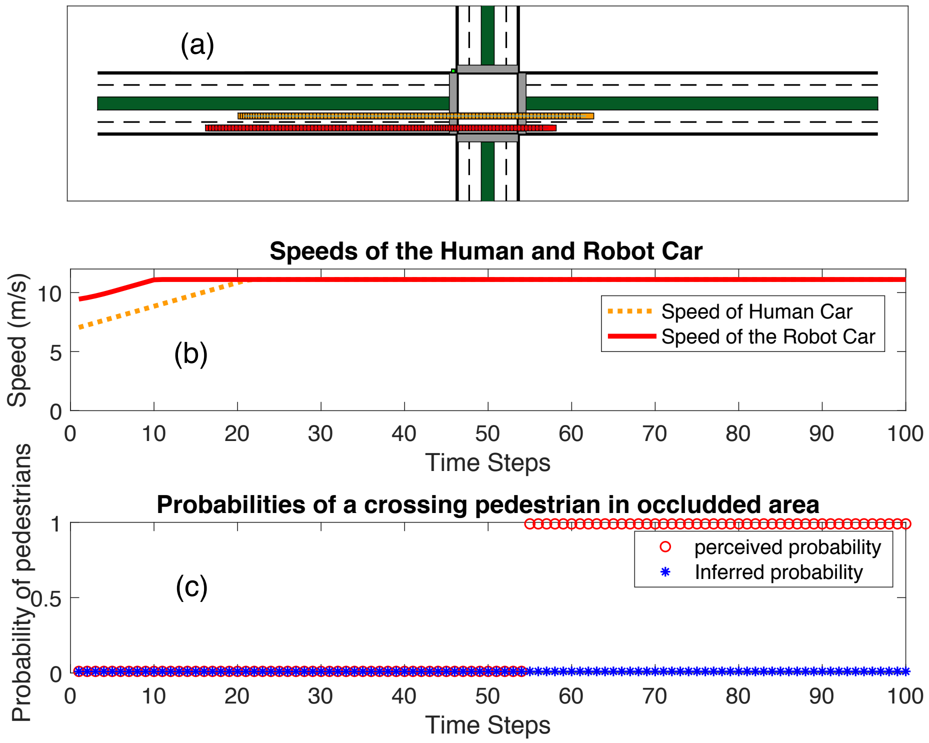


**Figure 6.6:** Simulation results with the proposed framework with non-crossing pedestrians in the occluded area: (a) yellow box - human driver; red box - robot car; green dot - pedestrian; (b) speeds of the robot and human cars; (c) the perceived and inferred probabilities of pedestrians

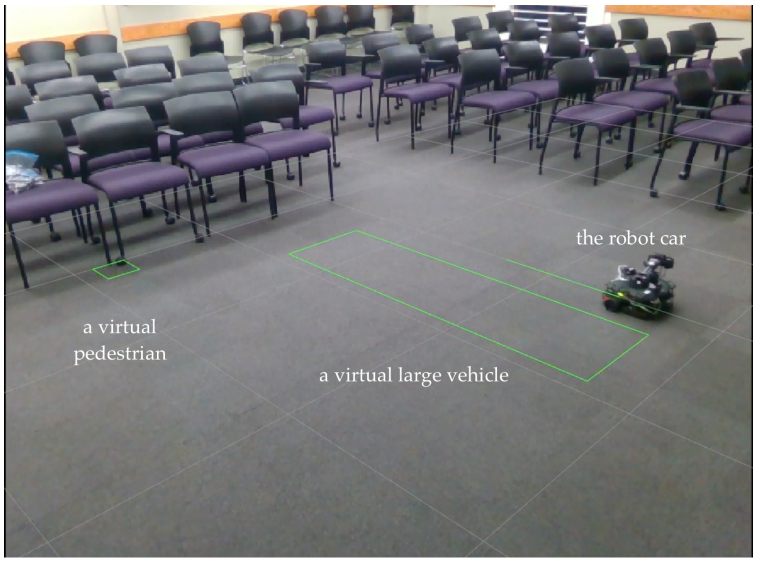
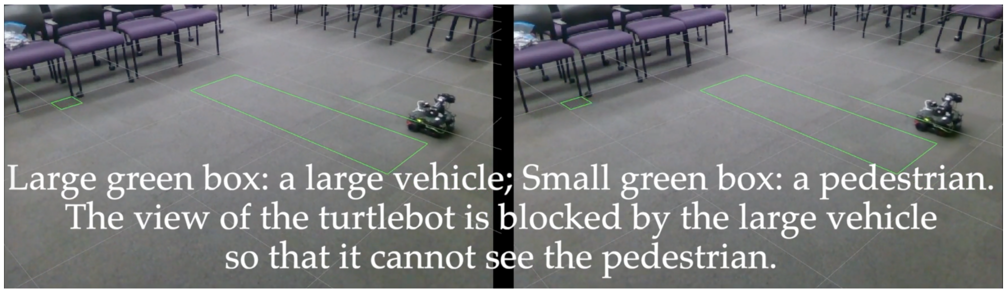
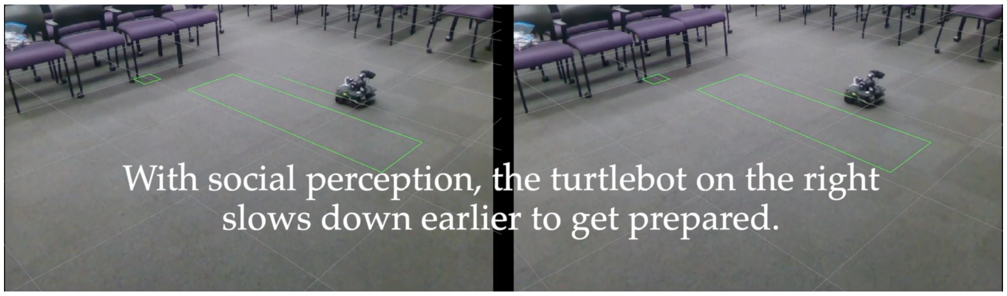


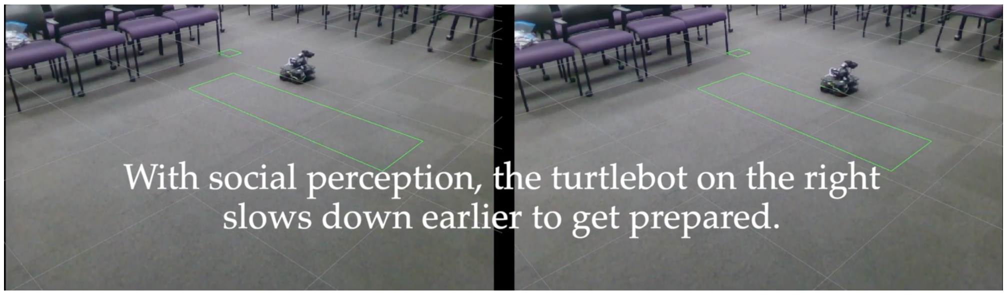
**Figure 6.7:** The experiment setup: the TurtleBot3 serves as a robot car, the small green box as a pedestrian trying to cross the street, and the large green box as a large vehicle which is blocking part of the view for the robot car.


Large green box: a large vehicle; Small green box: a pedestrian.
The view of the turtlebot is blocked by the large vehicle
so that it cannot see the pedestrian.

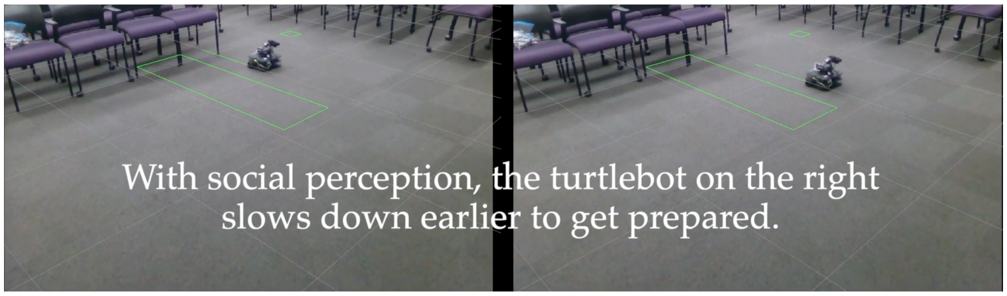
**(a)** t=0s


With social perception, the turtlebot on the right
slows down earlier to get prepared.

**(b)** t=10s


With social perception, the turtlebot on the right
slows down earlier to get prepared.

**(c)** t=14s, the TurtleBot3 with social perception stopped earlier than the one without


With social perception, the turtlebot on the right
slows down earlier to get prepared.

**(d)** t=18s

**Figure 6.8:** The experiment results with the TurtleBot3 and virtual pedestrian and vehicles: left - without social perception, right - with social perception.

# Chapter 7

# Behavior Planning Under Policy Uncertainties

Imagine that you are driving towards a narrow bridge with another car from the opposite direction. A bridge is narrow enough such that only one car can pass at a time. The two cars are driving at similar speeds and are away from the entrances from each side at similar distances. Suppose that you know exactly what the other driver's care about, can you accurately predict the his/her behavior? Unfortunately, the answer is no. There are multiple reasons behind. A major one lies in the interaction between you and the other driver. While you are predicting his/her behaviors, he/she is predicting yours. Even though we can utilize the multiple methods discussed in earlier chapters to learn as well as possible the reward functions, we have assumed that the other driver has full access to what you want to do in the future and explicitly utilized it as *a priori* knowledge during the learning and online inference process. Such assumption, however, is too strong to satisfy in practice.

As a matter of fact, the other driver might not pay attention to you, or he/she may interpret your behavior via a simple rule-based model such as maintaining current actions, or he/she is assuming that you are cooperative or aggressively non-cooperative. Moreover, their models are not fixed, but rather time-varying as the interaction goes on. An inattentive human driver might become attentive as you are getting closer, and a driver who believes you as an aggressive driver might change his/her mind as he/she observes your behaviors.

In this chapter, we aim to tackle the above-mentioned uncertainties regarding interactive policies and design a reliable behavior generation policy for the autonomous systems.

## 7.1 Introduction

We model the two-agent interaction problem based on two-player games. Many researches have utilized the game-theoretic setting to model the interaction between human and robots. For instance, Li et al. have conducted a series of work on game-theoretic decision making for autonomous vehicle control, assuming a Stackelberg policy is adopted in the interaction process. In [78] and [141], they built simulation scenarios with additional user inputs representing human drivers to find out the levels of human model in games, i.e., at what levels human perform games with other agents. He found that in most scenarios, human drivers are level-I game participants, namely, they consider the responses of others when they evaluate specific actions. Fisac et al.[34] also adopts the Stackelberg Game to model the dynamic game theoretic interactions. To make the problem solvable in real time, they partitioned the planning task into the long-term strategic planning problem with simplified dynamics and full information structure and short-term tactical planning problem with full dynamics and a simplified information structure. A highway scenario demonstration is shown. Besides, in [154, 153], the authors also adopted Stackelberg game theory to a driver behavior model in a merging situation. In [138], with V2V communication, the authors presented a lane-changing model based on the two-person non-zero-sum non-cooperative game under complete and incomplete information. In[148], the authors proposed a nonlinear receding horizon game theoretic planner using Nash equilibria and tested the algorithm using racing cars in simulations.

Most of the studies, however, are conducted in simulation with synthetic data. There is yet no answer to an important question: what policies do most human take during interaction? What *a priori* distributions should we assume when we adopt game-theoretic settings?

*Our key insight in this chapter is that human are flexible and uncertainty in terms of the game policy they adopt. Wrong assumptions on the policy can lead to interactive robotic actions that not only confuse human, but also are dangerous.*

Motivated by this, in this chapter, we propose to use online Bayesian inference to estimate the most probable game policy of human based on real human driving data. Moreover, based on the estimates, we design a robust policy for the robots to be compatible with uncertainties of the human's game policy.

## 7.2 Problem Statement

In this section, we consider the interactive behavior between two vehicles: the ego vehicle (denoted by $(\cdot)_{\text{ego}}$) and the other [1] vehicle (denoted by $(\cdot)_{\text{other}}$) .

[1]While extension of our formulation and solution to N players is well defined (and relatively straightforward) in theory, in practice the computation grows exponentially with the number of interacting vehicles. It is a fundamental open challenge. We thus limit the scope of this work to pairwise interactions.

For a given vehicle, we use $s$ to represent the state variable, $\hat{\gamma}$ for the predicted action, $\gamma_{gt}$ for the ground truth action and $\tilde{\gamma}$ for all the possible actions. The dynamics of the joint state $x^t \in \mathcal{X} \subset R^n$ of the vehicles in the world, which we assume to be fully observable [34] , are

$$\gamma^{t+1} = f(\gamma^t, s^t_{ego}, s^t_{other}) \tag{7.1}$$

where $\gamma^t, s^t$ denote, respectively, the action and state of the vehicle at time $t$.

Similarly, we assume that both agents generate their behaviors using the MPC framework. Namely, assuming that the individual finite-horizon cost functions are given as

$$C_{ego}(\gamma^t_{ego}) = \sum_{k=0}^{N-1} c(s^t, \gamma^{t,k}_{ego}, \hat{\gamma}^{t,k}_{other}; \theta_{ego}), \tag{7.2}$$

at time step $t$, both agents find the optimal sequence of actions that minimize the cost, execute the first action and repeat the process at the next time step $t+1$. Note that in (7.2), $\theta_{ego}$ characterizes the preference of the ego vehicle. We select $N = 5$ with a time interval $\triangle t = 0.2$s.

In practice, $\gamma^t, s^t$ are all continuous. However, to facilitate computation of the games, we discrete them. For the action space, i.e., the acceleration space, into a list with choices as $a = [-2, -1, 0, 1]$.

## 7.3 Monte Carlo Tree Search with Different Game Policies

### 7.3.1 Monte Carlo Tree Search

Monte Carlo Tree Search (MCTS) is a heuristic-based search algorithm. It has been widely used solve many game-theoretic problems such as the Go game in [125].

Based on the receding horizon algorithm, if the horizon is $n$ steps, we need to build a tree with depth of $2n+1$. Specifically, the root node represents the initial state of the two vehicles, with the odd levels for the decisions of the ego car and even levels for that of the other car. At each pair level, the two vehicles make decisions simultaneously, as you can see in Fig. 7.1.

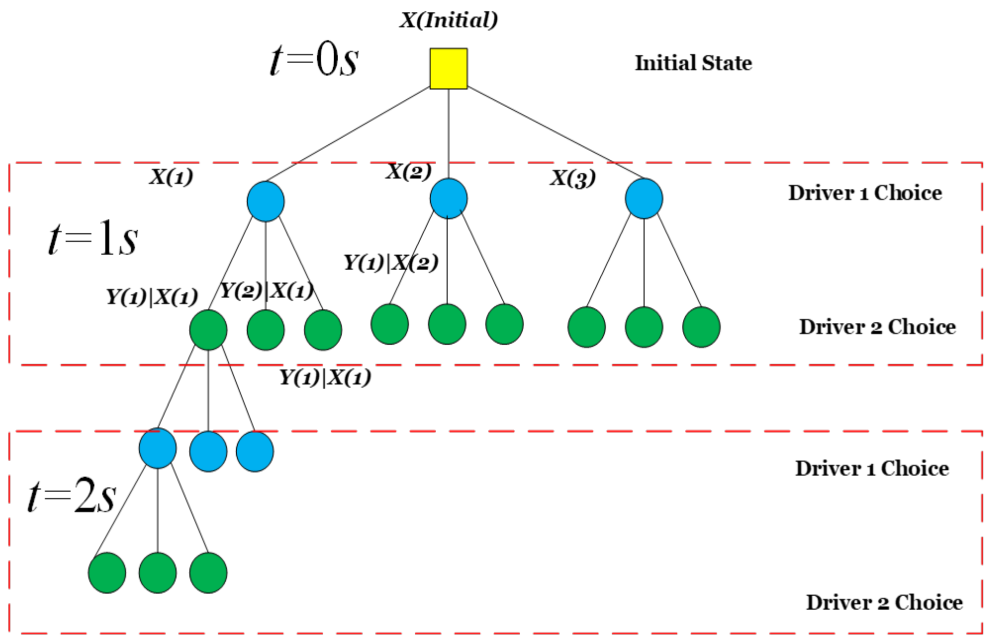


**Figure 7.1:** A illustration of the search trees to solve a finite-horizon optimization problem: the yellow root node represents the initial state of the two vehicles, and blue nodes are the actions by driver 1 and green nodes are for driver 2. Each paired layer in the red dotted box is assumed to happen simultaneously.

Given the cost function in (7.2), we can design the cost function for the tree search as follows:

$$C_{MCTS}(\gamma) = C(\gamma) + \theta \cdot \sqrt{\frac{n}{\log(N)}} \tag{7.3}$$

where the original cost function is augmented by a heuristic. Note that in (7.3), $n$ denotes the visit times of a certain children node, and $N$ denotes that of its parent node. $\theta$ is a weighting factor balancing between the exploration and exploitation. The MCTS algorithm we utilized is shown in Algorithm 2.

## 7.3.2 Five Interactive Policies

In this work, we consider five different interactive policies that the human might follow. The first three are theoretic policies whose equilibrium types are, respectively, Nash (non-cooperative), Stackelberg (non-cooperative) and Pareto (cooperative). The remaining two are incomplete-information approaches including a naive rule-based model and a ignoring policy which depicts the situation when the target human does not pay attention to the other agent.

To be more specific, Nash and Stackelberg are for non-cooperative games. Recent work finds that Stackelberg model can have a better representation of the decisions making process while driving. Pareto, on the other hand, is for cooperative games. It is

```
Input: $(s_t^{ego}, s_t^{other})$: Joint state of two vehicles at time $t$
       $N$: number of iteration
       $m$: number of time-step within consideration
Output: $\gamma_{optimal}^t$ : Optimal motion in the next time step
 1: Initialize the root node $n_0$ using the $(s_t^{ego}, s_t^{other})$
 2: Initialize the number of search $k = 0$
 3: while $k < N$ do
 4:     Set the current node $n$ using the root node $n_0$
 5:     Initialize the search depth $i = 1$
 6:     while $i < 2m + 1$ and all children nodes of $n$ expanded do
 7:         Select the children node $n_{child}$ using (7.3)
 8:         $n \leftarrow n_{child}$
 9:         $i = i + 1$
10:     end while
11:     if $i < 2m + 1$ and not all children of $n$ expanded then
12:         Explore the randomly selected children node $n_{child}$
13:         $n \leftarrow n_{child}$
14:         $i = i + 1$
15:     end if
16:     while $i < 2m + 1$ do
17:         Randomly Select $n_{child}$
18:         $n \leftarrow n_{child}$
19:         $i = i + 1$
20:     end while
21:     Backpropagate and update the visit times and the accumulated cost along the
        search route till the root node
22:     $k = k + 1$
23: end while
24: Output the child node of the root node with the largest visiting time as the optimal
    motion $\gamma_{optimal}^t$
```

**Algorithm 2:** A General Monte Carlo Tree Search Algorithm

widely adopted in centralized control such as V2X-enabled autonomous driving scenarios. The rule-based model can be as simple as assuming the other agent maintaining the current actions. For instance, for autonomous vehicles, it is commonly used to assume that a driver will maintain his/her speed or acceleration during the preview horizon. We simplify the notations for those five policies as: Nash, Stackelberg, Pareto, Constant, Ignore.

### Nash Equilibrium

In game theory, the Nash equilibrium is a solution for a non-cooperative game where no player can gain more utilities by changing only their own strategy [98]. Hence, we need to find out the action sequences for both the ego and the other agent which, respectively, minimize their cost functions. The solutions should satisfy

$$\gamma_{\text{ego}}^{t,*} = \arg\min_{\gamma_{\text{ego}}^{t}} \sum_{k=0}^{N-1} c(s^t, \gamma_{\text{ego}}^{t,k}, \gamma_{\text{other}}^{t,*}; \theta_{\text{ego}}) \tag{7.4}$$

$$\gamma_{\text{other}}^{t,*} = \arg\min_{\gamma_{\text{other}}^{t}} \sum_{k=0}^{N-1} c(s^t, \gamma_{\text{ego}}^{t,*}, \gamma_{\text{other}}^{t,k}; \theta_{\text{other}}) \tag{7.5}$$

The search process with MCTS is described in Fig. 7.2 and detailed algorithm is given in Algorithm 2.

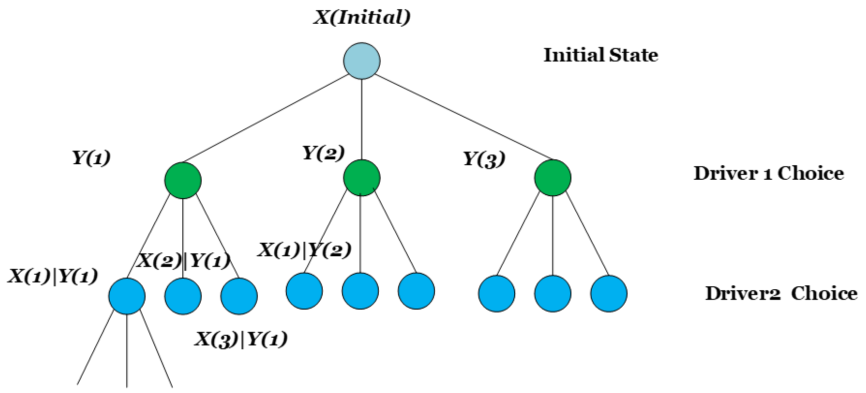


**Figure 7.2:** The MCTS process for a Nash equilibrium in a two-player game: the light-blue root node represents the initial state of the two vehicles, and green nodes are the actions by driver 1 (ego) and blue nodes are for driver 2 (the other).

### Stackelberg competition

The Stackelberg leadership model is a strategic game in which one of the players is a leader which moves first, and the other player is a follower which takes actions once he/she observes the actions from the leader [126]. In this work, we denote the ego vehicle as the leader, and the other car as the follower. With Stackelberg model, the two players make decisions sequentially: leader first and then the follower.

More specifically, in order to choose the optimal action for the leader X at time step $i$, we need to find the expected response of the follower Y under a certain choice of $\gamma_k(X_t)$, where $k$ denotes all the possible actions of X. Namely,

$$\gamma_{opt}(Y_{i+1}|\gamma_k(X_i)) = \arg_{\gamma(Y_{i+1})}\min \sum_{t=0}^{N-1} c_{follower}(\gamma_k(X_i), \gamma(Y_{t+1})) \tag{7.6}$$

Then, based on the best response of Y for each possible action of X, we can find the best decision of X as follows:

$$\gamma_{opt}(X_i) = \arg_{\gamma(X_i)}\min \sum_{t=0}^{N-1} c_{leader}(\gamma(X_t), \gamma_{opt}(Y_{i+1}|\gamma_k(X_t)) \tag{7.7}$$

Based on this idea, we summarize the MCTS algorithm in Algorithm 3 and Fig. 7.3.

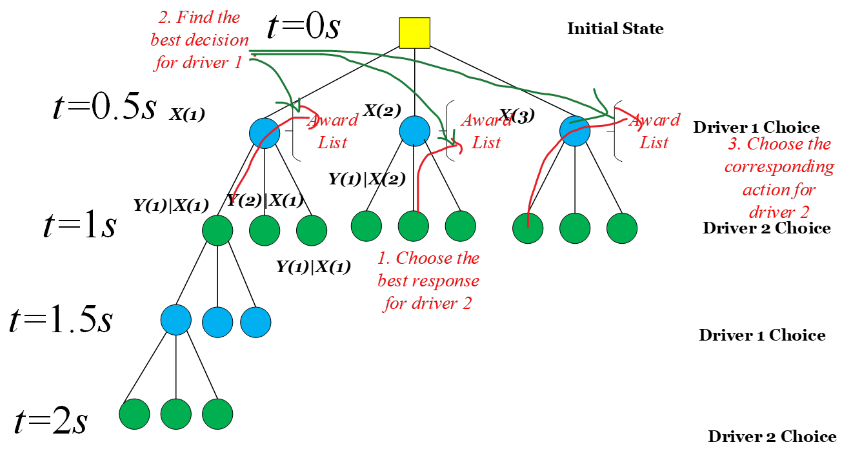


**Figure 7.3:** The MCTS process for a Stackelberg equilibrium in a two-player game: the yellow-square root node represents the initial state of the two vehicles, and blue nodes are the actions by the leader and green nodes are for the follower.

## Pareto Optimality

Pareto efficiency or Pareto optimality is a state of allocation of resources from which it is impossible to reallocate so as to make any one individual or preference criterion better off without making at least one individual or preference criterion worse off. In our case, we view two players equally important, which means the target cost function should be:

$$(\gamma_{opt}(X), \gamma_{opt}(Y)) = \arg_{\gamma(X),\gamma(Y)}\min[\sum_{t=0}^{N-1} c_{ego}(\gamma(X), \gamma(Y)) + \sum_{t=0}^{N-1} c_{other}(\gamma(X), \gamma(Y))] \tag{7.8}$$

```
Input: $(s_t^{ego}, s_t^{other})$: Joint state of two vehicles at time $t$
       $N$: number of iteration
       $m$: number of time-step within consideration
Output: $\gamma_{optimal}^{t}$ : Optimal motion in the next time step
 1: Initialize the root node $n_0$ using the $(s_t^{ego}, s_t^{other})$
 2: Initialize the number of search $k = 0$
 3: while $k < N$ do
 4:   Set the current node $n$ using the root node $n_0$
 5:   Initialize the search depth $i = 1$
 6:   while $i < 2m + 1$ and all children nodes of $n$ expanded do
 7:     if odd($i$) then
 8:       Find the optimal responses of the follower under all the possible actions of
          the leader using (7.6)
 9:       Choose the optimal decision $n_{child}$ for the leader using (7.7)
10:     else
11:       Choose the corresponding optimal response $n_{child}$ for the follower in (7.6)
12:     end if
13:     $n \leftarrow n_{child}$
14:     $i = i + 1$
15:   end while
16:   if $i < 2m + 1$ and not all children of $n$ expanded then
17:     Explore the randomly selected children node $n_{child}$
18:     $n \leftarrow n_{child}$
19:     for Every children node $n_{child}$of $n$ do
20:       Randomly select the children node till the end
21:       Back propagate the visiting time and cost
22:     end for
23:     $i = i + 1$
24:   end if
25:   while $i < 2m + 1$ do
26:     Randomly Select $n_{child}$
27:     $n \leftarrow n_{child}$
28:     $i = i + 1$
29:   end while
30:   Backpropagate and update the visit times and the accumulated cost along the
      search route till the root node
31:   $k = k + 1$
32: end while
33: Output the child node of the root node with the largest visiting time as the optimal
    motion $\gamma_{optimal}^{t}$
```

**Algorithm 3:** Search Strategy for Stackelberg Policy

Hence, the depth of the tree will reduce to half, but the number of nodes within a single layer will double in a two-player game. The MCTS process for Pareto solution is shown in Fig. 7.4.

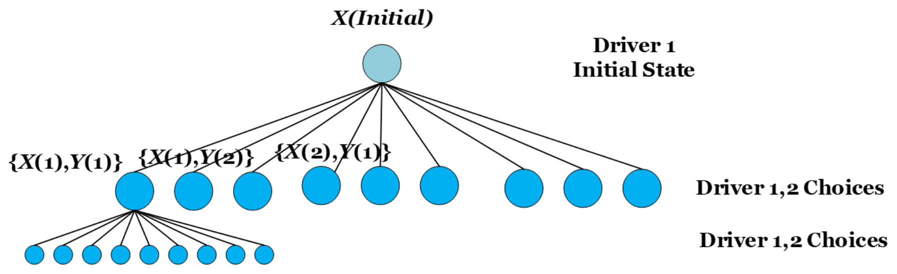


**Figure 7.4:** The MCTS process for Pareto equilibrium: starting from the initial condition represented by the light-blue root node, the two players are equally treated. At each layer, the action space is augmented by the actions from both agents.

## Constant-Action Policy

Using autonomous vehicles as an application example, the constant action we assume is that the vehicles will maintain its current speed, i.e., the acceleration is zero. Hence, with such assumption, our decision space only contains the ego vehicle. The depth of the search tree will reduce by half and the number of nodes in each layer remain the same with the action space for a single agent. The MCTS algorithm is described in Fig. 7.5.

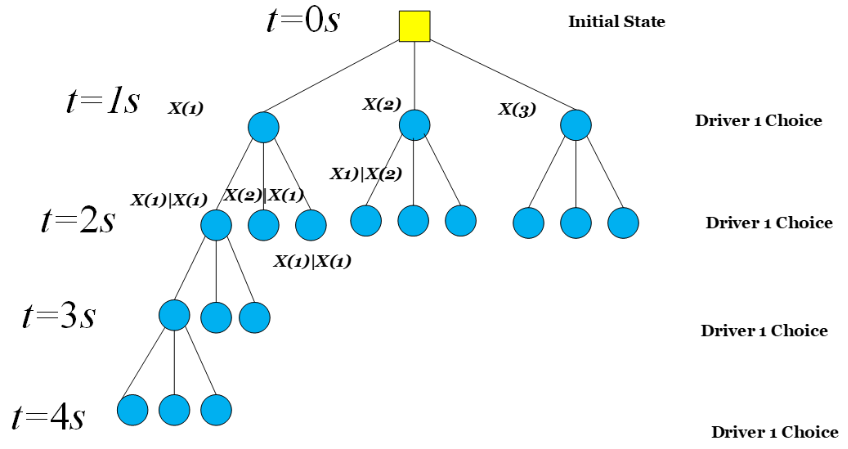


**Figure 7.5:** The MCTS process for constant-speed and ignoring policies: starting from the initial condition represented by the yellow-square root node, only the action space of the ego vehicle is explored in each layer.

**Ignore**

The ignoring policy is introduced in this section to represent the scenarios when a human driver is not paying attention to other agents that he/she is supposed to interact with. We assume that under this policy the information of the other driver is blocked, which means that the ego vehicle can't receive the state information of the other vehicle. Thus the cost function for the ego vehicle should only include terms defined on his own actions, as shown in (7.9). The MCTS algorithm is the same as in the "constant-action" policy shown in Fig. 7.5.

$$C_{ego}(\gamma_{ego}^{t}) = \sum_{k=0}^{N-1} c(s^{t}, \gamma_{ego}^{t,k}; \theta_{ego}) \tag{7.9}$$

## 7.4 Motion Planning with Online Bayesian Inference for Game Policies

To find out which policies a human are more likely to adopt during interaction, we adopt Bayesian inference to update the probability of each policy based on new observations at each step. $\pi_i (i = 1, 2, 3, 4, 5)$ denotes each of the five policies that the agent would adopt. Here we assume that the initial probability of each policy is the same, namely $P(\pi_i) = \frac{1}{5}(i = 1, 2, 3, 4, 5)$. At each time step, the probability for the i-th policy $P(\pi_i)$ is updated as follows:

$$P^{t}(\pi_i | a_t) \propto P^{t-1}(\pi_i) \cdot P(\gamma_t | \pi_i) \tag{7.10}$$

where $\gamma_t$ denotes the action that a human would adopt at time step t. Based on the principle of Maximum Entropy, the posterior probability $P(\gamma_t | \pi_i)$ is given by

$$P^{t}(\gamma_t | \pi_i) = \frac{e^{-Q^{*}(\gamma_t, s_t)}}{\sum_{\tilde{\gamma}} e^{-Q_i^{*}(\tilde{\gamma}_t, s_t)}} \tag{7.11}$$

where $\tilde{\gamma}_t$ denotes all the possible actions the driver would adopt at time step t, This is a typical Q-value inference where $Q^{*}$ denotes the cost to go given a specific action and the current state. To find $Q^{*}$ under each different game policies, we run the MCTS algorithms discussed in Section 7.3.2.

Here we modify the original MSCT algorithm based on the posterior information and show how to conduct the Bayesian Inference based on game theoretic approaches in Algorithm 4.

Once the policy beliefs are obtained, we integrate such information into the motion planning algorithm for the robot systems so that an expected utility under the game policy uncertainties can be maximized, as shown in (7.12).

$$\gamma_{opt}(X) = \arg_{\gamma(X)} \min \sum_{i=1}^{5} \sum_{t=0}^{N-1} P(\pi_i) \cdot c(\gamma(X), \gamma_i(Y)) \tag{7.12}$$

```
Input: $s^{ego}, s^{other}$:Joint state of two vehicles
Output: $P(\pi_i|a_t)$: The probability of adopting each policy
 1: Covert the $X-Y$ coordinate to $l-s$ coordinate.
 2: Compute the collision point and transfrom the origin of the $l-s$ coordinate to that
    point.
 3: Extract $s^{ego}, s^{other}$ within the interaction period
 4: Initialize time $t=1$.
 5: while $l_t^{ego} > 0$ and $l_t^{other} > 0$ do
 6:     for $\pi_i (i = 1, ..., 5)$ do
 7:         Compute posterior probability $P^t(a_t|\pi_i)$ using the cost value of children nodes
            using $(s_t^{ego}, s_t^{other})$ in Algorithm 2.
 8:     end for
 9:     Update the prior probability $P^t(\pi_i|a_t)$ in (7.10)
10:     $t = t + 1$.
11: end while
```

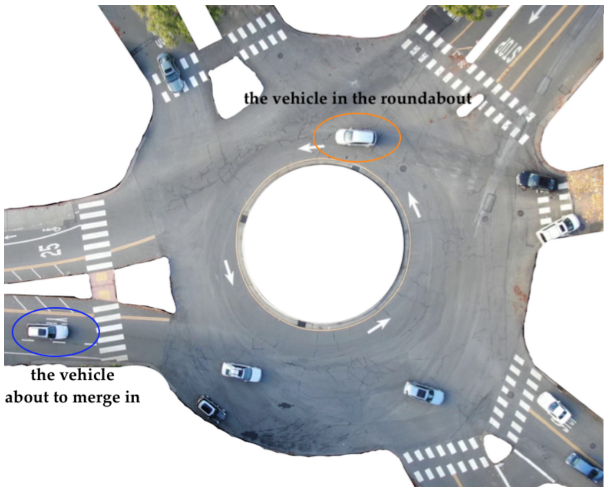
**Algorithm 4:** The Bayesian Inference Algorithm

With such policy uncertainty-aware motion planning strategy, we can design safer autonomous systems. Moreover, if the human switch policies, the robot can efficiently identify that and adapt its behaviors accordingly.

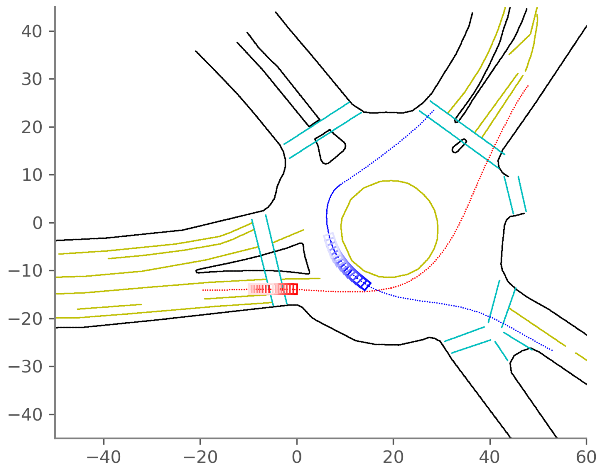
## 7.5 A Case Study

### 7.5.1 The roundabout Scenario

We demonstrate the effectiveness of the proposed policy-aware motion planning algorithm on a case study in autonomous driving. The interactive scenario we consider is a roundabout merging from the INTERACTION dataset [159, 158], as shown in Fig. 7.6(a). We focus on the interactive agents where one player is trying to merge into the roundabout, and the other player is already in the roundabout, passing by the entrance where the first player comes from. Hence, they have potential conflict or merging points on their future paths. We collected 253 pairs of such interaction trajectories where each trajectory contains a sequence of the vehicle's states and actions including $x-y$ coordinates, speeds, yaw angles and accelerations (one example is shown in Fig. 7.6(b)). Similar to what we have done in Section 5.4 in Chapter 5, we converted the trajectories from $x-y$ coordinates to coordinates in Frenet frame $(s-d)$ based on the identified reference paths in Fig. 5.3. Furthermore, for each interaction pairs, we set the conflict or merging point as the common origin point along the longitudinal direction $(s)$. Before they arrive at the conflict or merging point, their longitudinal coordinates are positive. Once the point is passed, their longitudinal coordinates become negative.



**(a)** The roundabout map and interactive vehicles



**(b)** An example of the extracted trajectories for interactive vehicles

**Figure 7.6:** The roundabout scenario in INTERACTION dataset we focused in this chapter and an exemplar pair of the interactive trajectories: (a) we care about the two interactive vehicles in the orange and blue circles where one is in the roundabout and other other is about to merge in. They have at least one conflict point or merging point on their future paths; (b) shows an exemplar pair of such interactive trajectories.

Utilizing the map and collected human driving data, we conducted two studies. First, we try to answer our motivation question - what policies do most human take during interaction? To find out the answers, we run the Bayesian policy inference algorithm in presented in Section 7.4 for each vehicle in the pairs and record how the game policies involve. The second study is the verification of the proposed policy-aware motion planning strategy. We run simulations on the roundabout map and test the performance.

## 7.5.2 Statistics of the Game Policies for Human

### The Cost Functions for Drivers

To run the algorithms in Section 7.3, we need to have access to human's cost functions, which is practically not possible. One might argue that we can use inverse reinforcement learning or inverse optimal control to learn the cost functions. However, given the unknown policies, inverse reinforcement learning or inverse optimal control will not work since they need such information as prior knowledge. If we make assumption on the policies and conduct inverse reinforcement learning or inverse optimal control, the learned cost functions will be biased and cannot be used to infer the correct policies. Hence, to avoid significant biases in the cost functions, we only consider the must-have features for driving in the assumed cost functions which are the speed feature and collision-avoidance feature. At each time step, the cost $c(\gamma_t)$ is defined as

$$c(\gamma^t) = w_1 \cdot |v(t) - v_{des}| + w_2 \cdot \varphi\left(x(t)\right) \varphi\left(y(t)\right) \lambda\left(x(t), y(t)\right) \tag{7.13}$$

where $|v(t) - v_{des}|$ quantifies the speed deviation from the desired speed $v_{des}$, and $\varphi\left(x(t)\right) \varphi\left(y(t)\right) \lambda\left(x(t), y(t)\right)$ measures the collision penalty (safety-related) of two vehicles. $w_1$ and $w_2$ are, respectively, the weights for the speed and the safety terms. In this work, we set $w_1 = 0.05$ and $w_2 = 50$. $x(t)$ and $y(t)$ represent the states for the two interacting vehicles. All the states are defined in the Frenet frame. The definition of $\varphi(x)$ and $\lambda(x, y)$ are given in (7.14) and (7.15):

$$\varphi(x) = \begin{cases} x, & \text{if} \quad 0 < x < K, \\ 0, & \text{else} \end{cases} \tag{7.14}$$

$$\lambda(x, y) = |K - |x - y|| \tag{7.15}$$

The hyper parameter $K$ is a safety distance term. It can be chosen based on our degree of emphasis on safety. In our experiment, we recommend $K \in [5, 10]$. The desired speed $v_{des}$ is calculated based on the speed limit, the geometry of the path and human's acceptance range for lateral accelerations:

$$v_{des} = \text{clip}(\sqrt{\frac{1.4}{|\kappa|}}, 0, v_{limit}) \tag{7.16}$$

where $\kappa$ denotes the curvatures of the reference curve, and $v_{limit}$ is the speed limit. In this environment, we set it as 25mph as posted in the real map.

### Statistic Results

Based on the cost functions defined in (7.13), we infer the game polices for each of the agent in the collected interaction trajectory pairs based on Algorithm 4. Figure 7.7 shows one exemplar results where we can see that the target human driver first tends to be cooperative, pursuing a Pareto equilibrium in the interaction. As interaction goes by, the driver becomes more aggressive and switches towards a Nash equilibrium.

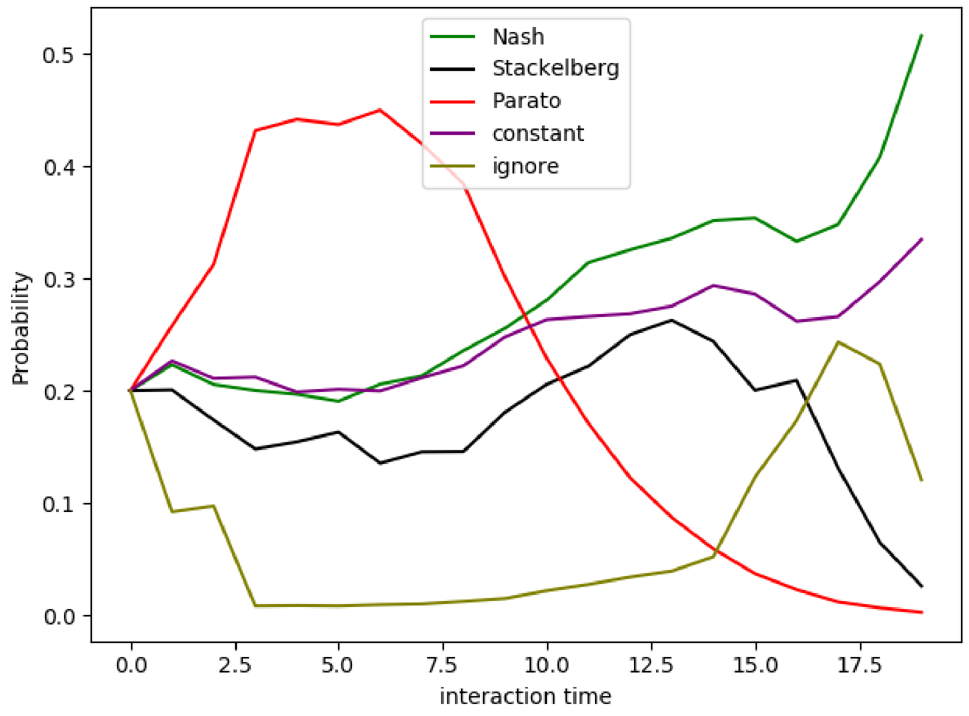


**Figure 7.7:** An exemplar evolution of the game policy on a human driving trajectory: the target human driver first tends to be cooperative, pursuing a Pareto equilibrium in the interaction. As interaction goes by, the driver becomes more aggressive and switches towards a Nash equilibrium.

For all the collected interaction pairs, we record such evolution of their game policies. We calculate two measures: dominance of policy and policy switching frequency.

- Dominance of policy: this measure quantifies how long each policy is serving as a dominant policy. As shown in Fig. 7.7, there are two dominant policies - the Pareto and Nash. We also considering the duration period for each policy in this measure, i.e., the dominating policies will be weighted by their dominant periods. Essentially, we record the dominating time period for each policy in each interaction trajectories.

- policy switching frequency: this measure counts how many times a human switches the policies during one interaction. Such a measure can give us hints on how frequency human drivers switch policies in interactive driving. For instance, in the result shown in Fig. 7.7, the target human switched two dominating policies.

Results on the “dominance of policy” are shown in Fig. 7.8. We can see that in most cases, human drivers are not interacting intensively, i.e., they are running the “ignoring” policy. When they are interacting, they tends to be more cooperative than competitive since the “Pareto” policy dominates more than “Nash”, “Stackelberg” and “Constant” policies. Such results match our observations with courteous driving in human drivers: human tends to be cooperative and courteous to each other during interaction, in most scenarios.

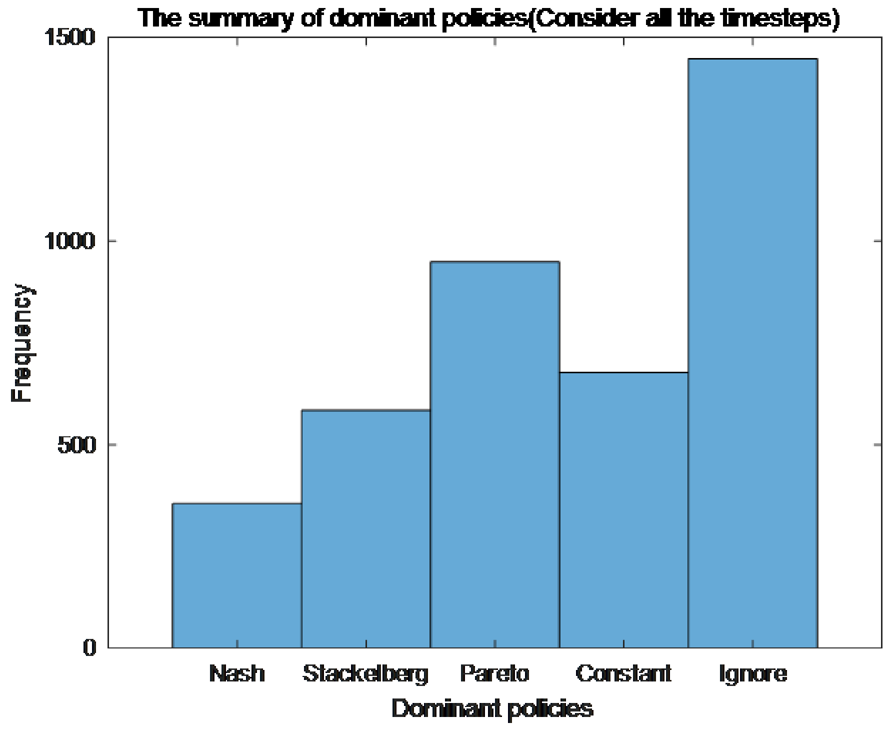


**Figure 7.8:** Results on the "dominance of policy": in most cases, human drivers are not interacting intensively, i.e., they are running the "ignoring" policy. When they are interacting, they tends to be more cooperative than competitive since the "Pareto" policy dominates more.

Figure 7.9 shows the results on "policy switching frequency". Among the 253 sets of data, in most cases, human switched only once or twice for the game policies. There is a significant drop between three times and more. Such results can serve as important prior knowledge when we design robots' behavior. We should not let the robot switch policies too frequently, for instance, more than 3 times during one interaction. Moreover, to model the human behaviors, we can also assume that human will not frequently change their policies given the observed results in Fig. 7.9.

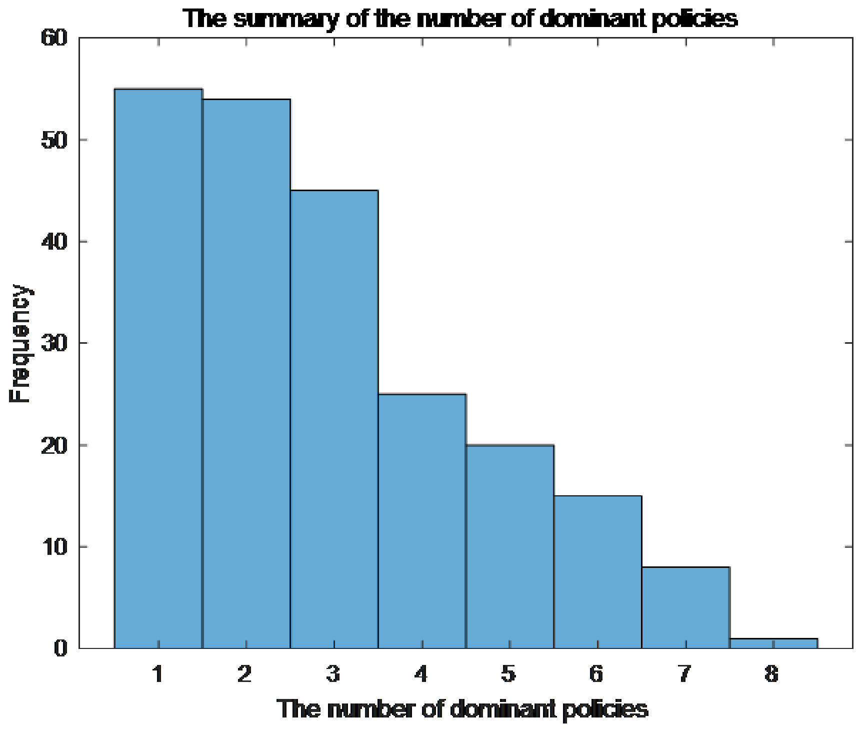


**Figure 7.9:** Counts on the number of dominating policies during one interaction over the collected dataset: in most scenarios, human will not switch for more than 3 policies.

### 7.5.3 Motion Planning with Policy Uncertainties

Based on the inference possibilities, we run the policy-aware motion planning strategy in simulation on the roundabout map. We let the robot be the vehicle trying to merging into the roundabout with exactly the same initial conditions as the real data, and keep the other vehicle's trajectory untouched. To quantify how human-like our proposed motion planning strategy is, we again utilize two measures.

- Discrete interaction results: for the discrete interaction results, we use the order of the two vehicles passing their conflict or merging point. If the passing order using our motion planning algorithm is the same with ground truth, we claim the algorithm can generate human-like behavior in the sense that it does not change the distribution of the discrete interaction results. The results are shown in Fig. 7.10. It matches the ground truth distribution quite well.

- Continuous interaction results: for the discrete interaction results, we use mean square error along the longitudinal direction which is defined as follows:

$$MSE = \frac{1}{n}\sum_{t=1}^{n}(s_t - \tilde{s_t})^2. \tag{7.17}$$

where $n$ is the number of test trajectories, and $s_t$ and $\tilde{s}_t$ are, respectively, the ground truth longitudinal coordinate at time $t$ and the re-generated one using our proposed motion planning algorithm. The results are shown in Fig. 7.11. We can see that in most scenarios, the $MSE$ between our re-generated trajectories and the ground truth trajectories are small, which means that our proposed policy-aware motion planning algorithm can generate human-like behaviors.

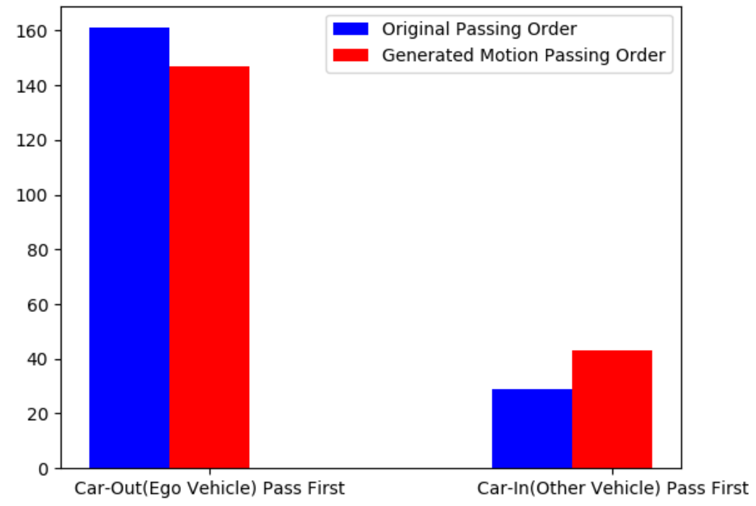


**Figure 7.10:** The distribution of passing orders over the dataset for two interacting vehicles: blue represents the ground truth passing order and red is the result with the generated motions.

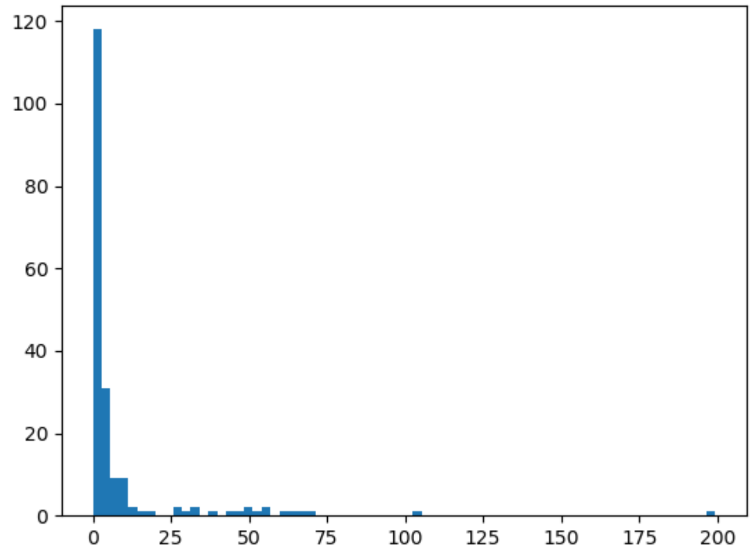


**Figure 7.11:** The histogram of the mean square error along the longitudinal direction between the generated trajectories of the robot vehicle and the ground-truth trajectories

## 7.6 Chapter Summary

Motivated by finding appropriate assumptions for the game policies in game-theoretic bebavior design, in this chapter, we designed a scheme to infer the game policies of agents during interactions based on Bayesian inference. Built upon that, we developed a policy-aware motion planning strategy which can help generate safe actions in the presence of policy uncertainties in interactions. To evaluate the effectiveness, we performed two studies: one on the human's game policy in human driving and the other on the human-likeness of the proposed motion planning algorithm. Through the first study, we found that human tends to be cooperative in most scenarios, and they do not switch policies too often. Moreover, the motion planning results showed that the policy-aware motion planning algorithm can well preserve the distributions of the interactive results in the collected dataset.

The work in this chapter is step towards more human-like behavior design for autonomous systems. Beyond the perception uncertainties in Chapter 6, we address the problem of uncertain game policies in interaction. The work can be further extended in many directions. For instance, more substantial statistical results on more driving scenarios can be obtained to provide more prior knowledge for the research community focusing on autonomous driving. We can also explore strategies to integrate the inference of game policies with reward learning to eliminate the influence from biased cost function designs.

# Chapter 8

# Safety-Enhanced Imitation Learning

In this chapter, we will discuss the problem of efficiently generate behaviors for autonomous system using advanced technologies such as learning and model-based optimization. Safety and efficiency are two key elements for planning and control in autonomous systems. Theoretically, model-based optimization methods, such as Model Predictive Control (MPC), can provide such optimal policies. Their computational complexity, however, grows exponentially with horizon length and number of surrounding agents. This makes them impractical for real-time implementation, particularly when nonlinear models and variety of uncertainties are considered, as addressed in Chapter 6 and Chapter 7. To enable a fast and approximately optimal behavior policy, we propose a safe imitation framework, which contains two hierarchical layers. The first layer, defined as the *policy layer*, is represented by a neural network that imitates a long-term expert policy via imitation learning. The second layer, called the *execution layer*, is a simplified model-based optimal controller that tracks and further fine-tunes the reference trajectories proposed by the *policy layer* to avoid collisions. Moreover, to reduce the distribution mismatch between the training set and the real world, Dataset Aggregation is utilized so that the performance of the *policy layer* can be improved from iteration to iteration.

We address the problem using autonomous driving as an application example, but the proposed framework can be adapted to other human-robot interaction systems.

## 8.1 Introduction

In recent years, autonomous driving has attracted a great amount of research efforts in both academia and industry for its potential benefits on safety, accessibility, and efficiency. Typically, autonomous driving systems are partitioned into hierarchical structures including perception, decision-making, motion planning and vehicle control (see [23, 24]). Among them, planning and control are two core challenging problems that are responsible for safety and efficiency. They should

1. comprehensively consider all possible constraints regarding safety and feasibility

as much as possible based on the perceived environment information and reasonable prediction of other road participants' behaviors;

2. generate optimal/near-optimal maneuvers that provide good driving qualities such as smoothness, passengers' comfort and time-efficiency;

3. solve the problem within limited runtime to timely respond to rapid changes in surrounding environment.

Simultaneously satisfying the above requirements can be difficult and many planning and control approaches have been proposed [40]. They can be categorized into five groups: i) graph-search-based, such as $A^*$(see [33]), ii) sampling-based (e.g., RRT, RRT*, see [65, 58]), iii) interpolating-curve-based [15, 80], iv) optimization-based [86, 79], and v) learning-based approaches [107, 17, 64]. Groups i) and ii) are space-discrete in the sense that they discretize the state space into grids/lattices or sampled nodes and search for solutions that build feasible connections among them. Consequently, the resulting paths/trajectories are not continuous but jerky, and a subsequent smoother is necessary for realistic implementation. Approaches in iii) can mostly generate smooth paths, but it is hard to guarantee its optimality, not even locally. Besides, without temporal consideration, dealing with moving obstacles can be time-consuming using such approaches. Optimization-based approaches, on the other hand, formulate all possible cost functions and constraints in a uniform manner as constrained optimization problems with continuous state and action spaces. A popular example utilizes Model Predictive Control (MPC). At each time step, the optimal control actions are generated by solving a constrained optimization problem (in general highly nonlinear and non-convex). A practical issue is that the computational load grows exponentially with the horizon length and the number of obstacles, which makes it hard to be implemented in real time, particularly when a long horizon is preferred for persistent feasibility and safety. In order to guarantee the realtimeness, several methods have been proposed. For instance, in [168], the sequential optimization is terminated at pre-determined maximum runtime. In [167, 160] the unified optimization problem is decomposed into two or more sub-problems with linearized vehicle models and constraints. Both methods sacrifice optimality for computational efficiency.

On the other hand, "End-to-End" learning approaches in group v) are increasingly attracting attentions in recent years due to their powerful representations of the environment and fast forward computation in test phases. Typically, such learning-based approaches take raw images as inputs, and directly output the driving actions or a driving policy net by training a deep neural network via supervised learning [17] or deep reinforcement learning [150, 41]. As pointed by [25], however, the outputs of such "End-to-End" networks are hard to yield any guarantees for feasibility, safety and smoothness. Also, such "End-to-End" structure is hard to incorporate *a priori* knowledge on vehicle models, which makes their training not only data-hungry, but also time-consuming, i.e., hours/days of training time on multiple GPUs. This makes the "End-to-End" learning system hard to adapt to new situations. This can be dangerous in practice since real-time deviations from the training set/simulation environment can lead to unpredictable error propagation and cause severe consequences if the training set/simulation environment cannot be augmented by realistic test results.

In this chapter, we propose a hierarchical structure that combines the advantages of both learning- and optimization-based methods via safe imitation mechanism. The framework consists of two layers. The first layer, defined as the ***policy layer***, is represented by a neural network trained via imitation learning. The inputs of the network is a set of highly representative features that describe the environment information, and the output is an instructive trajectory imitating a long-term expert driving policy. The second layer, called the ***execution layer***, is an approximated constrained optimal controller that aims to generate safe and feasible actions efficiently by fine-tuning the instructive trajectories from the ***policy layer***. We explore two approximation schemes: formulating it as a short-horizon optimal controller or as a long-term approximated optimal controller. *Our key point is to replace a time-consuming long-horizon online optimizations with offline training and efficient online generation and adjustment*. In such sense, not only the long-term optimality is preserved, but also *a priori* knowledge on vehicle models can be fully utilized for better tracking and control performance. Moreover, Data Aggregation (DAgger) process is utilized to consistently improve the performance of the policy layer.

## 8.2 The Hierarchical Structure

Figure 8.1 shows the overview of the proposed planning and control framework for autonomous driving. It consists of three hierarchical modules: perception, decision-making, and planning and control. At each time step, the perception module detects the surrounding environment via on-board sensors (e.g., GPS, LiDAR), and yields measurements/estimates of all necessary states of the ego vehicle such as in-map location, orientation, velocity and its relative positions and velocities to all other visible road participates (static or moving). Based on the perception results and pre-defined driving tasks, the decision-making module will set the reference lane to instruct the next-level planning and control.

We focus on the planning and control module. To satisfy all the three requirements discussed above, the planning and control module consists of two layers in test phase[2]: the ***policy layer*** and the ***execution layer***. First, based on the driving decisions (e.g., target lane) and the perception results, a set of highly representative features are extracted as inputs of the ***policy layer*** that is trained to yield trajectories imitating that of a long-term expert driving policy (for instance, such a policy can be directly obtained via long-term MPC or sampling approaches such as RRT*). Then, with the imitated reference trajectory, the ***execution layer*** computes the optimal control actions (steering angles and

[2]Note that in the training phase, no execution layer is need and the training of the policy layer is a typical behavior cloning process.

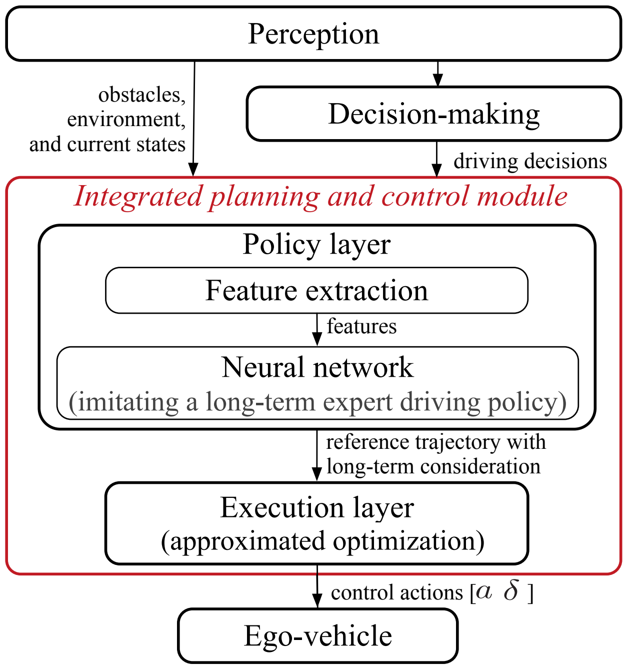

Perception
obstacles, environment, and current states
Decision-making
driving decisions
Integrated planning and control module
Policy layer
Feature extraction
features
Neural network
(imitating a long-term expert driving policy)
reference trajectory with long-term consideration
Execution layer
(approximated optimization)
control actions [$a$ $\delta$ ]
Ego-vehicle

**Figure 8.1:** The overall hierarchical structure

accelerations) that considers model constraints, feasibility and obstacle avoidance so that the ego vehicle can be safely driven. To assure that the planning and control module is environmentally responsive, we adopt the re-planning scheme, i.e., at each time step, only the first control actions from the execution layer are sent to the actuators to drive the ego vehicle.

## 8.3 The Design of the Execution Layer

To efficiently generate safe and feasible control actions, we explore two approximation schemes in the ***execution layer*** layer: 1) formulating it as a short-term constrained trajectory tracking problem and 2) formulating it as an approximated long-term constrained trajectory tracking problem, and more specifically, a quadratic programming problem. Both are using the framework of MPC with two goals: 1) track as much as possible the trajectories generated by the policy layer and 2) make safe adjustments when necessary to guarantee collision avoidance.

### 8.3.1 Approximation Scheme I: A short-Term Optimal Controller with Constraints

In this subsection, we introduce the first approximation scheme: formulating it as a short-term trajectory tracking problem with constraints. Take the driving scenario in

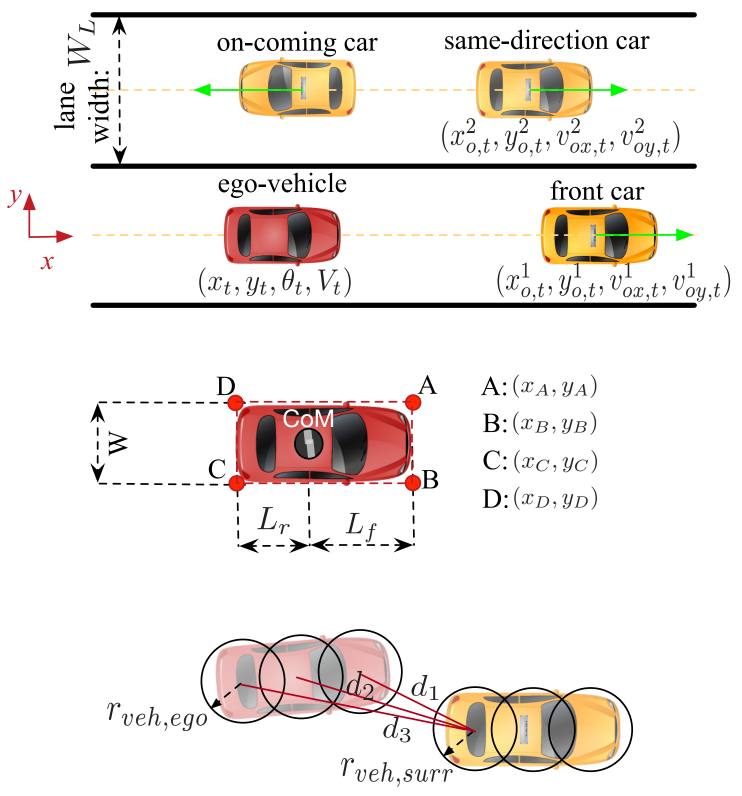


**Figure 8.2:** An example driving scenario for short-term MPC

Fig. 8.2 as an example. Assume that at each time instant $t$, states of the ego-vehicle are represented by $z_t=[x_t, y_t, \theta_t, V_t]^T$, where $(x_t, y_t)$ is current car position, $\theta_t$ is the yaw angle and $V_t$ is the speed. Similarly, states of all surrounding cars are denoted by $(x^i_{o,t}, y^i_{o,t}, v^i_{ox,t}, v^i_{oy,t})$ $(i=1, 2, \cdots, n)$. Control actions of the ego vehicle include the longitudinal acceleration and steering angle, i.e., $u_t=[a_t, \delta_t]^T$. Define the horizon length in execution layer as $N_e$, and at time $t$, the within-horizon predicted ego-vehicle's states and control variables are respectively $z^p_{t,k+1}=[x^p_{t,k+1}, y^p_{t,k+1}, \theta^p_{t,k+1}, V^p_{t,k+1}]^T$ and $u^p_{t,k}=[a^p_{t,k}, \delta^p_{t,k}]^T$ with $k=0, 1, 2, \cdots, N_e-1$, and the within-horizon surrounding cars' states are denoted by $(x^{p,i}_{o,t,k+1}, y^{p,i}_{o,t,k+1}, v^{p,i}_{ox,t,k+1}, v^{p,i}_{oy,t,k+1})$ for $i=1, 2\cdots, n$.

*Remark 1*: In this chapter, we assume that the prediction of future states of surrounding vehicles is available. Typically, they can be predicted using simple rule-based behavior models (e.g., zero-input or constant velocity assumptions) or advanced prediction algorithms as discussed in Chapters 2–5.

### Objective Function

Suppose that the first $N_e$ trajectory points given by the policy layer are $[(x^r_{t,1}, y^r_{t,1}), (x^r_{t,2}, y^r_{t,2}), \cdots, (x^r_{t,N_e}, y^r_{t,N_e})]$, then a quadratic cost function is defined as

$$J^e_t = \sum_{k=1}^{N_e} \begin{bmatrix} x^e_{t,k} \\ y^e_{t,k} \end{bmatrix}^T W_e \begin{bmatrix} x^e_{t,k} \\ y^e_{t,k} \end{bmatrix} + u^T_{t,k} W_u u_{t,k} + \triangle u^T_{t,k} W_{\sigma u} \triangle u_{t,k} \tag{8.1}$$

where $[x^e_{t,k}, y^e_{t,k}] \triangleq [x^p_{t,k} - x^r_{t,k}, y^p_{t,k} - y^r_{t,k}]$ represents the tracking error and $\triangle u_{t,k} \triangleq [a_{t,k} - a_{t,k-1}, \delta_{t,k} - \delta_{t,k-1}]$ is the derivatives of the control inputs. $W_e \in \mathcal{R}^{2\times 2}$, $W_u \in \mathcal{R}^{2\times 2}$ and $W_{\sigma u} \in \mathcal{R}^{2\times 2}$ are all positive definite matrices that tune the weights of tracking error, control effort and comfort level of the passengers.

### Constraints

The short-term tracking problem comprehensively considers all constraints from the system kinematics feasibility (nonlinear equality constraints), dynamics feasibility and collision-free safety (inequality constraints).

- *Kinematics feasibility*: Bicycle model [113] is adopted in this work to express the kinematic model of the ego vehicle, i.e., for $k=0, 1, \cdots, N_e-1$ within the preview horizon:

$$\begin{aligned} x^p_{t,k+1} &= x^p_{t,k} + V^p_{t,k} \cos(\theta^p_{t,k} + \tan^{-1} \frac{L_r \tan \delta^p_{t,k}}{L}) dt & (8.2) \\ y^p_{t,k+1} &= y^p_{t,k} + V^p_{t,k} \sin(\theta^p_{t,k} + \tan^{-1} \frac{L_r \tan \delta^p_{t,k}}{L}) dt & (8.3) \\ \theta^p_{t,k+1} &= \theta^p_{t,k} + \frac{V^p_k \tan \delta^p_{t,k}}{L} \cos(\tan^{-1} \frac{L_r \tan \delta^p_{t,k}}{L}) dt & (8.4) \\ V^p_{t,k+1} &= V^p_{t,k} + a^p_{t,k} dt & (8.5) \end{aligned}$$

  and $x^p_{t,0} = x_t, y^p_{t,0} = y_t, \theta^p_{t,0} = \theta_t, V^p_{t,0} = V_t$. $L = L_r + L_f$ is the total car length as shown in Fig. 8.2 and $dt$ is the discrete time interval.

- *Dynamics feasibility*: The dynamics feasibility is guaranteed via constraints from the G-G diagram, as shown in Fig. 8.3, where $a^+_{max} < a^-_{max}$ represent, respectively, the maximum acceleration input and deceleration input. Therefore, the feasible region (shadow part in Fig. 8.3) can be expressed in terms of decision variables as:

$$\left( \frac{x^p_{t,k+2} - 2x^p_{t,k+1} + x^p_{t,k}}{dt^2} \right)^2 + \left( \frac{y^p_{t,k+2} - 2y^p_{t,k+1} + y^p_{t,k}}{dt^2} \right)^2 \leqslant (a^-_{max})^2 \tag{8.6}$$

$$a^p_{t,k} \triangleq a^p_{lon,t,k} \leqslant a^+_{max} \tag{8.7}$$

- *Safety constraints*: Safety is assured by constraining the ego-vehicle's configurations (position and orientation) and its distances to surrounding cars. As shown in Fig. 8.2, we decompose each of the vehicle into $m$ circles of radius $r_{veh}$, which are laid out equidistantly along its longitudinal axis as explained in [168]. To assert collision free, we require that the distances from the ego vehicle's circles to all surrounding vehicles' circles to be larger than $r_{veh,ego} + r_{veh,surr} + \epsilon$ where $\epsilon$ is a pre-determined safety buffer distance between two cars. Mathematically, for $n$ surrounding vehicles, this brings in $N_e n m^2$ safety constraints

$$d_{t,k}^{i,j} \geqslant r_{veh,ego}^{j} + r_{veh,surr}^{i,j} + \epsilon, \tag{8.8}$$
$$i{=}1,2,\cdots,n, j{=}1,2,\cdots,m^2, k{=}0,1,\cdots,N_e-1.$$

Moreover, other safety constraints to ensure that the ego vehicles does not hit the curb are also considered. By limiting the corner points ($A, B, C, D$ in Fig.8.2) of the ego vehicle within the road boundaries, another $2N_e$ nonlinear inequalities are generated. Hence, the total number of safety constraints is $N_e n m^2 + 2N_e$.

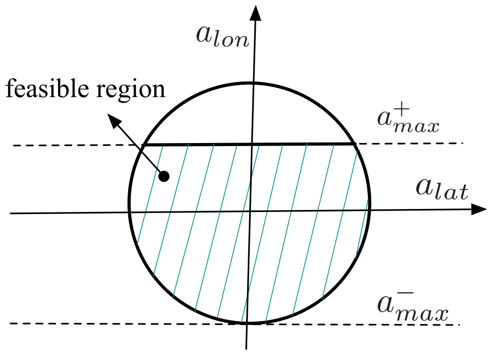


**Figure 8.3:** Acceleration boundaries from G-G diagram

In summary, the short-term time-varying optimal control problem in the execution layer at each time instant $t$ is given by:

$$\begin{aligned} P_t := & \min_{(z_{t,0}^p,\cdots,z_{t,N_e}^p,u_{t,0}^p,\cdots,u_{N_e-1}^p)} J_t^e \\ \text{s.t} \quad & z_{t,k}^p \in \mathcal{Z}, \forall k{=}0,1,\cdots,N_e \\ & u_{t,k}^p \in \mathcal{U}, \forall k{=}0,1,\cdots,N_e{-}1 \\ & \text{constraints Eqns. (8.2)-(8.8)} \end{aligned} \tag{8.9}$$

where $\mathcal{Z}$ and $\mathcal{U}$ are the bounded set of the ego vehicle's states and control variables. By solving this optimization problem online, we can get the optimal action sequence $u_{t,0}^p, \cdots, u_{N_e-1}^p$ at time $t$, execute the first action $u_{t,0}^p$ and re-plan at time $t+1$ in a manner of receding horizon control.

### Efficient Proximate Learning

When the execution layer is formulated as a short-term trajectory tracking problem subjected to constraints, we use an efficient proximate learning scheme in the imitation layer. Recall that in receding horizon control, at each time step $t$, although a sequence of trajectories/control actions are generated, only the first one will be applied to drive the system. Similar concept will be adopted in the imitation process. Suppose that the length of the long-term trajectory given by an expert policy is $N$, it makes more practical sense if the short-term optimal controller in the execution layer can track the first several reference points than the later ones. In fact, if the execution layer can track the first several reference points precisely, the influence of the remaining ones are ignorable. Hence, instead of imitating all $N$-length reference trajectories given by an expert policy, we introduce the concept of "proximate learning" in the policy layer when the execution layer is formulated as a short-term trajectory tracking problem. Namely, as shown in Fig. 8.4, the training loss is defined as a weighted sum of Euclidean distances between the network outputs and the reference expert trajectory only within the proximate area:

$$l = \sum_{[s_k^{net}, d_k^{net}]}^{\text{proximate area}} W_d \left(d_k^{net} - d_k^{ref}\right)^2 + W_s \left(s_k^{net} - s_k^{ref}\right)^2 \tag{8.10}$$

where $(s_k^{ref}, d_k^{ref})$ represents the expert trajectory in the Frenet Frame and $(s_k^{net}, d_k^{net})$ is the output of the network in the policy layer, which is later converted into $(x_t, y_t)$ coordinate for the execution layer to safely track. $W_d$ and $W_s$ are, respectively, the imitation weights along the lateral and longitudinal directions. In this case study, we set the proximate areas to contain 10 sequential points, i.e., equal to the horizon length $N_e$ in the execution layer.

Such configuration helps reduce the number of output nodes in the output layer of the network and consequently the network size, so that training time can be significantly reduced, which allows fast adaptation of the system via online DAgger, as discussed in Section 8.4-C. Moreover, it also reduces the horizon length for the following execution layer, i.e., reducing the overall computational load of the proposed framework.

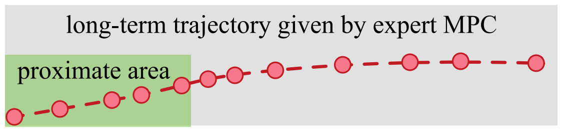


**Figure 8.4:** Illustration of the proximate learning concept

### 8.3.2 Approximation Scheme II: An Approximated Quadratic Programming Problem

The first approximation scheme in Section 8.3.1 can only assure short-term safety and collision avoidance. Hence, it may suffer from hard brakes in order to avoid collisions. Typically, we also would like a long horizon in the execution layer in order to generate more smoother maneuvers. Hence, in this subsection, we explore another approximation scheme in the execution layer: approximating the original long-term non-convex constrained optimization problem as a quadratic programming (QP) problem to improve the computation efficiency with the help of the long-term reference given by the policy layer.

The original long-term constrained optimization problem is the same as the one defined in (8.9) except that the horizon length will be much longer. To make the problem solvable in real time, we propose to approximate it via a QP problem utilizing the reference trajectory given by the policy layer. As shown in (8.9), in the optimization problem, there are constraints coming from the kinematic and dynamic models of the car and safety. For the kinematics and dynamics constraints, we linearize the kinematic and dynamic models at the given waypoint at each time step. For the safety constraints, we approximate the non-convex feasible set as convex hulls. Details are explained below.

Figure 8.5 shows an exemplar scenario for the robot car (the red car). The black lines represent the road curbs and the orange line is the lane separator. The green line and green dots represent the spatial-temporal reference waypoints given by the policy layer. The black dotted lines are perpendicular to the tangents of the reference trajectory at these waypoints. In Fig. 8.5(a), the white car is treated as a longitudinal obstacle. The lateral obstacles come from the lane boundaries. The semi-transparent white cars represent the future positions of the white car. In (b), the blue car is treated as a lateral obstacle.

- Longitudinal obstacles: with the given reference trajectory via the policy layer, we can directly detect whether the reference trajectory collides with longitudinal obstacles. For instance, as shown in Fig. 8.5(a), the green line is overlapping with future positions of the white car at the fifth time step. Hence, we can directly constrain the longitudinal position of the robot car at the fifth time step to be within the boundary represented by the purple line, i.e., the rear of the white car at the fifth time step. The same approach can be used in a similar way to handle cars behind the robot car.

- Lateral obstacles: for lateral obstacles, we linearize the boundaries of them by utilizing the reference trajectory. As shown in Fig. 8.5, at each time step, we can find the tangents of the reference trajectory and the feet of the perpendiculars on the boundaries (represented by the red dots in Fig. 8.5(a)). Through those feet of the perpendiculars, we can linearize the boundaries by lines that are parallel to the tangents of the reference trajectory (the red and blue lines in Fig. 8.5). Considering the constraints from each step, the non-convex feasible set is approximated via a combination of half spaces, i.e., a convex hull, as shown Fig. 8.5(b).

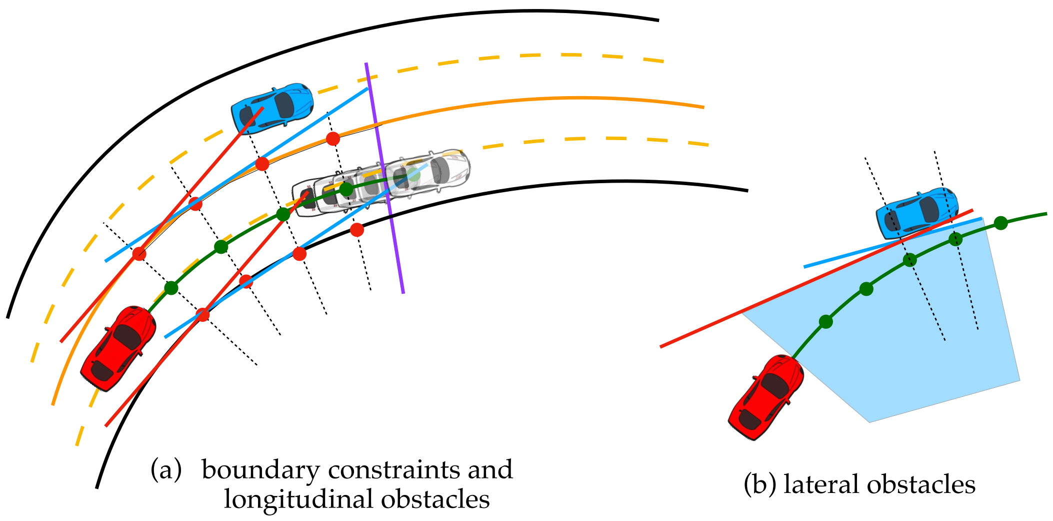


**Figure 8.5:** Illustration of the approximation for safety constraints: the red car is the robot/ego car, the blue car is an lateral obstacle, and the white car is a longitudinal obstacle. The green line and green dots represent the spatial-temporal reference waypoints given by the policy layer. The black dotted lines are perpendicular to the tangents of the reference trajectory at the waypoints. In (a), the red dots are the feet of the perpendiculars on the lane boundaries. The red and blue lines in both (a) and (b) are, respectively, parallel to the tangents of the reference trajectory at the first and second waypoints, and go through the feet of the perpendiculars on the lane boundaries and the other car's outline.

## 8.4 Design of the Policy Layer

As discussed above, as the preview horizon and the number of surrounding obstacles increase, the computational loads of direct optimization-based planning approaches increase exponentially, which makes them impractical in real-time online applications. In fact, even solving the safe tracking problem in Eqn. (8.9) with a quadratic cost function is not easy in realtime when $N_e$ is fairly large, not to mention an optimization-based planning problem with more complicated cost function that considers all necessary driving qualities. To address such time efficiency challenge while maintaining its optimality (to some extent), we proposed to design a policy layer to imitate an expert long-term driving policy as a fast online reference trajectory generator, and let the execution layer safely track the reference trajectory. Note that such an expert policy can be obtained via either offline optimization or other approaches (e.g., rapidly-exploring random tree (RRT) and RRT*) or from human driving data. In this section, details with respect to the imitation learning process will be discussed.

As for the input features, we explore two formats: numerical features and visual features. With numerical features, the input layer of the network is a vector, while for visual features, the input layer takes images as inputs.

## 8.4.1 Numerical Feature Extraction

For an efficient and compact network in the policy layer, feature selection is of key importance. The numerical features we selected can be divided into three groups at each time instant $t$:

- *on-map motion features*: ego-vehicle's current location and speed within the abstract map settings;
- *goal features*: ego-vehicle's lateral distance to the target lane set by the decision-making module at current time instant;
- *safety features*: ego-vehicle's current relative positions, orientations and velocities with respect to surrounding obstacles/cars.

### On-Map Motion Features

To effectively describe the *on-map motion features* on curvy roads, we re-express all the coordinates of vehicles in the Frenet Frame, as shown in Fig. 8.6. The position $(x_t, y_t)$ at time $t$ is translated to $(s_t, d_t)$ where $s_t$ is the longitudinal travelled distance along the road and $d_t$ is the lateral deviation from the lane center.

First the lateral distance of the ego-vehicle to curbs are given, namely, $f_t^{map,1} \triangleq d_{t,curb}^{+} - d_t$ and $f_t^{map,2} \triangleq d_t - d_{t,curb}^{-}$. Furthermore, we spatially discretize the proximate reference lane center, the coordinates of which are represented by $r_{unit} \triangleq [x_1^c, y_1^c, x_2^c, y_2^c, \cdots, x_j^c, y_j^c, \cdots, x_m^c, y_m^c]$. Then the current deviations of the ego-vehicle from the reference lane center can be represented by $d_t^y \triangleq [d_t^{y,1}, d_t^{y,,2}, \cdots, d_t^{y,j}, \cdots, d_t^{y,m}]$, where each element can be calculated by

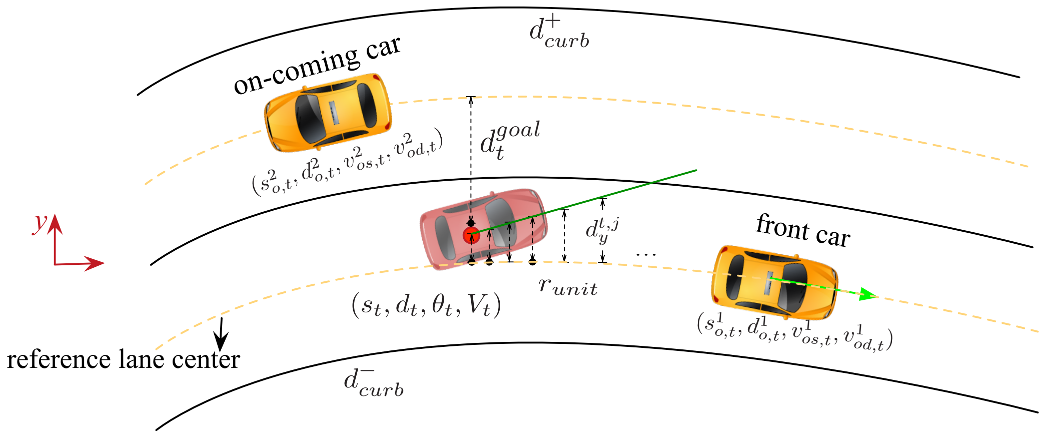
$$d_t^{y,j} = \mathsf{FF}(y_t + (x_j^c - x_t)\tan\theta_t - y_j^c), \forall j = 1, 2, \cdots, m. \tag{8.11}$$

where $\mathsf{FF}(\cdot)$ represents the function to convert from $x - y$ coordinate to Frenet Frame coordinate.

In this work, $m$=10 is set, i.e., ten on-map-motion-related features $f_t^{map,j+2} \triangleq d_y^{t,j}$ for $j$=1, 2, $\cdots$, 10 are generated.

As for the speed-related feature, it should indicate the current ego-vehicle's speed $V_t$ as well as its margin to the pre-defined speed limit. Hence, we define $f_t^{map,13} \triangleq V_t^{mar} = V_t - V_{max}$.

*Remark 2*: $r_{unit}$ is not necessarily evenly sampled in distance. Actually, noting the fact that closer proximity matters more, $r_{unit}$ can be sampled spatially with increasing inter-



$d^{+}_{curb}$
on-coming car
$d^{goal}_t$
$(s^2_{o,t}, d^2_{o,t}, v^2_{os,t}, v^2_{od,t})$
$y$
$d^{t,j}_y$
front car
$r_{unit}$
$(s_t, d_t, \theta_t, V_t)$
$(s^1_{o,t}, d^1_{o,t}, v^1_{os,t}, v^1_{od,t})$
reference lane center
$d^{-}_{curb}$

**Figure 8.6:** Definitions for feature selection

vals. Such configuration allows $r_{\text{unit}}$ to cover a longer distance, which helps improve the features' sensitivity to small yaw angle changes.

### Goal Features

Given the current goal position $d_t^{goal}$, as shown in Fig. 8.6, the goal feature is defined as

$$f_t^{goal} \triangleq d_t^{goal}. \tag{8.12}$$

### Safety Features

Considering the states of all surrounding vehicles, the following features are extracted, for $i=1, 2, \cdots, m$,

$$f_{t,i}^{safety} = \left[s_t - s^i_{o,t}, d_t - d^i_{o,t}, v_{t,s} - v^i_{os,t}, v_{t,d} - v^i_{od,t}\right]^T \tag{8.13}$$

where $v_{t,s}$ and $v_{t,d}$ are, respectively, the longitudinal velocity and the lateral velocity, and similarly, $v^i_{os,t}$ and $v^i_{od,t}$ are the velocities for the $i$-th surrounding vehicle. For $m$ surrounding vehicles, $4m$ features are generated.

All above defined features $f_t^{map}$, $f_t^{goal}$ and $f_t^{safety}$ generate a set of highly representative features $f_t = [(f_t^{map})^T, f_t^{goal}, (f_t^{safety})^T]^T$ and will be used as inputs of the neural network in the policy layer.

## 8.4.2 Visual Feature Extraction

The numerical features mainly work in relatively simple scenarios. For more complicated scenarios, it is hard to extract the numerical features. For instance, in real driving scenarios, the number of vehicles in the scene can change, and it is hard to adapt that with a fixed network structure using numerical features. Hence, in this subsection, we extend the feature set in the policy layer from vector-based features to visual features to make the policy layer more expressive. Four groups of features are included: the routing feature, the road mask feature, the dynamic obstacle feature and the velocity feature. All the features are converted as bird-view images with a size of $100 \times 100$, overlooking a space of $80m \times 80m$ in the map centered at the ego vehicle's current position. A sequence of feature sets will be generated along each trajectory with a down-sampling rate of five, i.e., 0.5s per frame. More details regarding each feature are given as follows.

#### The routing features

The routing features serve for similar purposes as the goal features in the simulation part. It encourages the ego vehicle to drive along the assigned routing. As shown in Fig. 8.7(a), the white pixels represent the ground-truth routing of the ego vehicle.

#### The road mask features

An example of the road mask feature is shown in Fig. 8.7(b). We represent all drivable areas via the white pixels and all other areas as black ones. The road mask feature can encourage the ego vehicle to drive within road boundaries.

#### The dynamic obstacle features

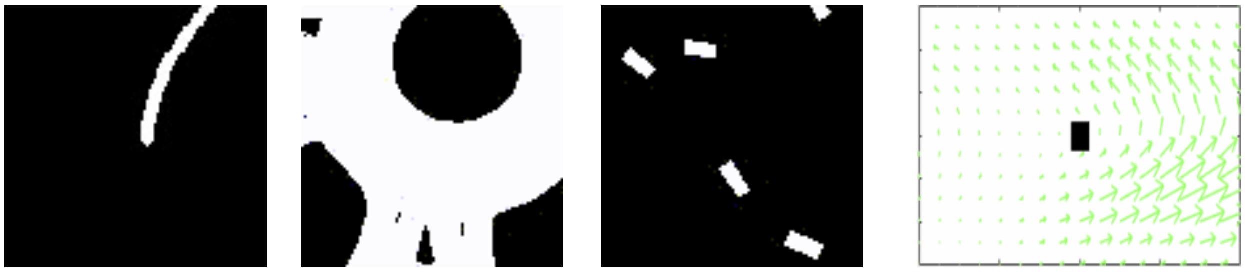
Similarly, all the dynamic objects such as other vehicles in the scene (within the $80m \times 80m$ region) are represented as white rectangles in the feature images, as shown in Fig. 8.7(c). It encodes information regarding the relative positions among other agents and the ego vehicle.

#### The velocity features

To represent the relative velocities between other agents and the ego vehicle as feature images, we adopt the Gaussian velocity field as in [53, 163]. As shown in Fig. 8.7(d), a Gaussian velocity field describes a joint Gaussian distribution of the relative velocities of all the agents with respect to the ego vehicle in the target region (within the $80m \times 80m$ region). Both velocity fields along the $x$ direction and $y$ direction will be generated as two channels in the input layer.

### 8.4.3 Loss

The output of the policy layer is a sequence of future waypoints with respect to the ego vehicle's current position, i.e., $(\hat{x}_t, \hat{y}_t), t = 1, 2, \cdots, N$ where $N$ is the length of the prediction horizon. In this experiment, we set $N = 6$, i.e., we predict the future trajectory within $6 * 0.5s = 3s$. Note that the $(x, y)$ coordinates are defined in Frenet



(a) the routing feature (b) the road-mask feature (c) the dynamic-obstacle feature (d) the Gaussian velocity field feature

**Figure 8.7:** The features in the policy layer to imitate human driving behavior: (a) the feature for the routing information, (b) the feature for the drivable area, (c) the feature for the dynamic obstacles, and (d) the feature for the Gaussian velocity field.

frame, with $y$ as the travelled distance along the longitudinal direction and $x$ as the lateral deviation from the reference path (i.e., the routing information). In the training process, we augment the imitation loss with other routing-related loss and collision-related loss to penalize wrong routings and collisions. Details regarding the losses are given as follows.

## Imitation loss

The imitation loss quantifies how much the predicted future positions deviate from the ground truth. Let $(x_t, y_t)$ be the ground truth coordinates, and $(\hat{x}_t, \hat{y}_t)$ be the predicted ones, then the imitation loss $L_{\text{imitation}}$ is:

$$L_{\text{imitation}} = \frac{1}{N}\sum_{t=1}^{N}\left[(x_t - \hat{x}_t)^2 + (y_t - \hat{y}_t)^2\right], \tag{8.14}$$

where $N$ is the length of the prediction horizon.

## Yaw angle correction loss

The yaw angle correction loss penalizes infeasible yaw angles. We bounds reasonable yaw angles within $[-\pi/4, \pi/4]$. Therefore, the yaw angle correction loss is:

$$L_{\text{yaw}} = \frac{1}{N}\sum_{t=1}^{N}\text{ReLU}\left(\frac{\hat{x}_t}{\hat{y}_t} - 1\right). \tag{8.15}$$

## Routing correction loss

The routing correction loss is defined to encourage the ego vehicle to drive along the ground-truth reference path. Suppose that there are $R$ waypoints representing the reference path, i.e., $[(x_1, y_1), (x_2, y_2), \cdots, (x_R, y_R)]$. For any predicted point with coordinates $(\hat{x}, \hat{y})$, we quantify its loss via a sum of distances to the reference path defined as

$$L_{\text{routing}} = \frac{1}{\sqrt{2\pi\sigma_r^2}} \sum_{i=1}^{R} sign(i) \exp\left\{ -\left(\frac{\hat{x} - x_i}{\sigma_r}\right)^2 - \left(\frac{\hat{y} - y_i}{\sigma_r}\right)^2 \right\}. \tag{8.16}$$

**Curb collision loss**

The curb collision loss is to penalize predicted future trajectories that collide with curbs. We also represent the curbs via a sequence of waypoints, i.e., $[(x_1, y_1), (x_2, y_2), \cdots,$ $(x_R, y_R)]$ where $R$ is the number of the waypoints for each segment of the curb. Then the curb collision loss is given by

$$L_{\text{curb}} = \sum_{i=1}^{R} \frac{sign(i)}{\sqrt{-(\hat{x} - x_i)^2/\sigma_r^2 - (\hat{y} - y_i)^2/\sigma_r^2 + \epsilon}}. \tag{8.17}$$

**Dynamic obstacle collision loss**

The dynamic obstacle collision loss is to penalize predicted future trajectories that collide with other dynamic agents in the scene. Suppose there are $M$ agents, and we represent each of them via its current position i.e., $[(x_1, y_1), (x_2, y_2), \cdots, (x_M, y_M)]$. Similar to the curb collision loss, the dynamic obstacle collision is given by

$$L_{\text{obstacle}} = \sum_{i=1}^{M} \frac{sign(i)}{\sqrt{-(\hat{x} - x_i)^2/\sigma_{x,i}^2 - (\hat{y} - y_i)^2/\sigma_{y,i}^2 + \epsilon}}. \tag{8.18}$$

**Final loss**

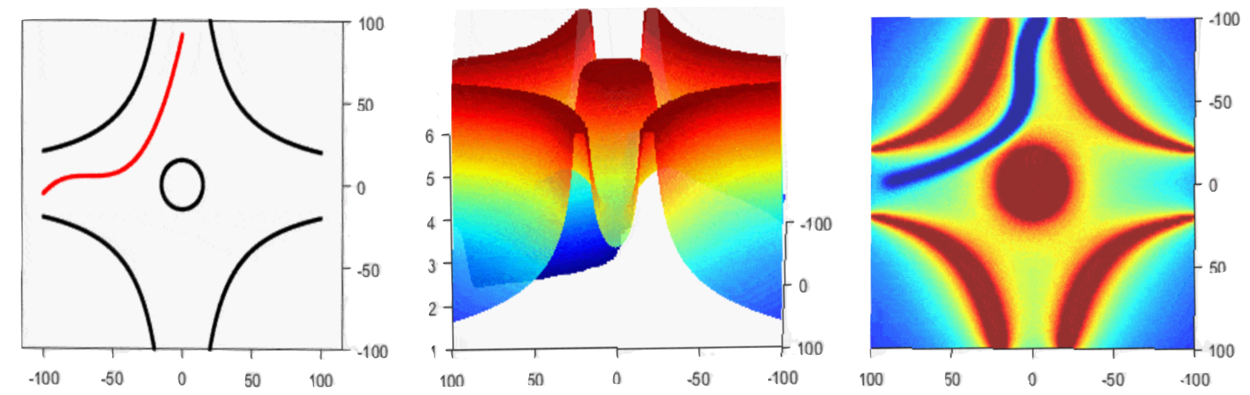
The final loss can be a combination of all the losses defined above, i.e.,

$$L_{\text{final}} = \omega_1 L_{\text{imitation}} + \omega_2 L_{\text{yaw}} + \omega_3 L_{\text{routing}} + \omega_4 L_{\text{curb}} + \omega_5 L_{\text{obstacle}}. \tag{8.19}$$

Figure 8.8 shows one example of the generated loss map where the black lines in (a) give the boundaries of the road, and the red line showing the reference path. (b) is the corresponding 3D loss map, and (c) is the projected 2D loss map. The redder the color, the higher the loss.

### 8.4.4 Learning via Sampled-DAgger Process

It is well known that, behaviour cloning itself is often not enough since it is never practical to generate a training set that satisfies the same states distribution as the real-world test data. Such distribution mismatch may make the learned policy biased and



(a) an example map (boundaries as black lines) and reference (red line)

(b) the generated loss map in 3D

(b) the generated loss map in 2D

**Figure 8.8:** An example of the loss map: (a) gives the map (black lines as boundaries) and the reference (the red line), (b) is the corresponding 3D loss map, and (c) is the projected 2D loss map. The redder the color, the higher the loss.

cause severe problems during execution, see, e.g. [115]. Once the encountered states slightly deviate from the training set (i.e., unfamilar features), the learned policy might yield wrong output, and such error will propogate and eventually fails the system.

One effective way to solve this is to use DAgger [115]. As an iterative algorithm, it adds all on-policy data into the training set so that, over iterations, the training set can cover most scenarios that the learned policy might encounter based on previous experience.

Motivated by this, a customized DAgger, defined as Sampled-DAgger, is proposed to improve the performance of the policy layer. As shown in Fig. 8.9, the process of the Sampled-DAgger are given as follows:

1. Gather an initial training set $\mathcal{D}_0$ by running a long-term expert planning algorithm $\pi_{expert}$ for randomly generated scenarios in simulation and train the network to yield an initial policy $\pi_0$.

2. Run the autonomous driving architecture shown in Fig. 8.1 with the learned policy $\pi_0$ in policy layer. Meanwhile, run $\pi_{expert}$ in parallel at a slow rate to periodically label the features with expert outputs. As shown in Fig. 8.9, $\pi_{expert}$ is running every M time steps, i.e., $T^{expert}_{interval}$=$Mdt$, which is set to be long enough to find the solution given by $\pi_{expert}$.

3. Compare the proximate loss (i.e., Eqn. (8.10)) of current policy $\pi_0(f_{t=kT^{expert}_{interval}})$ w.r.t the expert policy $\pi_{expert}$. If the normalized loss is larger than the pre-defined safety criterion, label the features $f_{t=kT^{expert}_{interval}}$ with corresponding expert outputs and push them into a new training set $\mathcal{D}'$. Once the size of $\mathcal{D}'$ reaches a pre-defined threshold, go to step 4).

4. Aggregate previous training set with $\mathcal{D}'$, i.e., $\mathcal{D} = \mathcal{D} \cup \mathcal{D}'$ and re-train the network to yield a new policy $\pi_{new}$ and set $\pi_0$=$\pi_{new}$.
5. Repeat 2) to 4) until the performance of the learned policy $\pi_0$ achieves similar performance as $\pi_{expert}$.

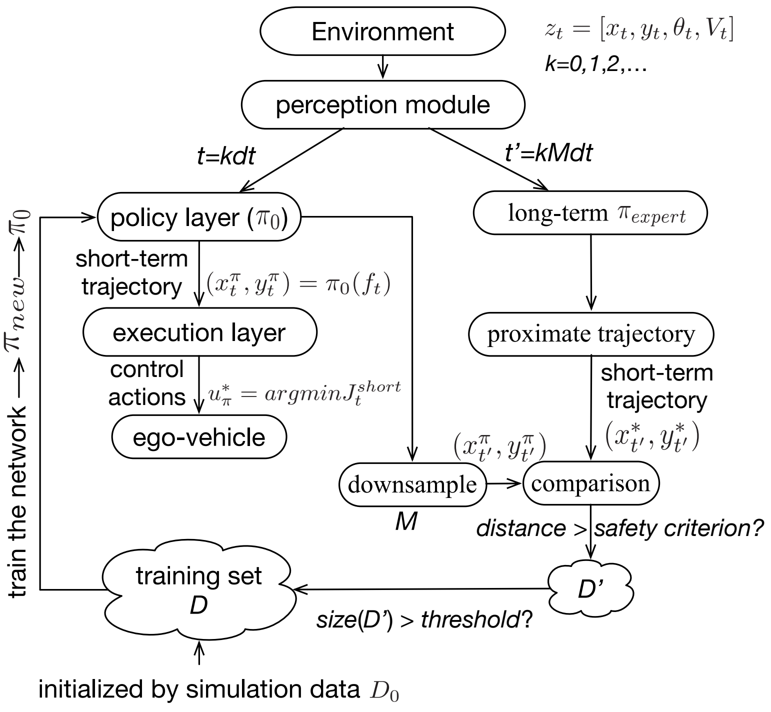


**Figure 8.9:** The Sampled-DAgger procedure

## 8.5 A Case Study for Highway Driving in Simulation

In this section, the proposed efficient planning and control framework is applied on a simple high-way driving scenario for verification. We consider two typical high-way driving behaviors with two surrounding vehicles ( $m = 2$ with one front car and another adjacent-lane car): 1) overtaking and 2) car following.

### Simulation Settings

The simulation environment is established based on a 1/10 scale ratio controlled (RC) car with a sampling period of dt=0.1s. The speed limit is set as 1 m/s for the RC car. In this scenario, we utilize the numerical features as discussed in Section 8.4.1 with a fully connected neural network in the policy layer. Detailed configuration of the network is given in Section 8.5. For the safe execution layer, we utilize the first approximation scheme, i.e., formulating it as a short-term trajectory tracking problem with constraints. A horizon length of $N_e$=10 is selected for solving the constrained optimization problem in (8.9). More detailed parameters for the simulation environment can be found in Section 8.5.

All simulations are performed using Julia on a Macbook Pro (2.5 GHz Intel Core i7) using the standard Ipopt solver.

**Table 8.1:** Parameters in the simulation environment

| layers | number of nodes | activation function |
|---|---|---|
| Input layer | 22 | sigmoid |
| hidden layer 1 | 50 | sigmoid |
| hidden layer 2 | 50 | sigmoid |
| output layer 2 | 20 | none |

**Table 8.2:** Parameters in the simulation environment

| $W_L$ | $L_r$ | $L_f$ | $W$ | $\epsilon$ |
|---|---|---|---|---|
| 0.38 (m) | 0.19 (m) | 0.21 (m) | 0.19 (m) | 0.02 (m) |
| $N$ | $N_e$ | $V_{max}$ | $a^+_{max}$ | $a^-_{max}$ |
| 30 | 10 | 1.0 (m/s) | 0.5 (m/s$^2$) | -1.0 (m/s$^2$) |

**Training Dataset**

We adopt a long-term MPC-based planning strategy to collect the training data with a preview horizon of N=30. Initially, the size of the training dataset $\mathcal{D}_0$ is 20k. For each iteration in the DAgger process, we run the trained policy with randomized vehicle states settings for 100 roll-outs/trajectories, where each rollout/trajectory period is 10s (i.e., 1000 points for dt=0.1s).

### 8.5.1 Performance for Overtaking Maneuver

Figure 8.10 and Fig. 8.11 show the results of an overtaking maneuver with the proposed hierarchical structure and a short-term MPC only. In this scenario, the ego vehicle (red) should bypass the slowly moving front car (blue) when the left lane is clear (left-lane car is also represented by blue rectangles). It can be seen that, in Fig. 8.10, the policy layer effectively imitated the expert long-term planning which enables the ego vehicle to accelerate and turn left at an early stage for a smoother and safe overtaking trajectory. When the yaw angle of the ego vehicle is relatively large, the policy layer brakes a bit to avoid possible collision with curbs. On the contrary, without the policy layer, the short-term MPC in Fig. 8.11 behaves "blindly" due to a too-short planning horizon: acceleration without long-prepared turning, sharp brakes and large steering angles when it is too close to the front car and finally repeated curb hitting and stopping.

## 8.5.2 Performance for Car Following

In car-following scenarios, no overtaking is allowed (this is set by the decision-making module considering environment settings, for instance, not enough safe margin for overtaking or adjacent lanes are blocked). To utilize the same policy layer as in overtaking maneuveur, a virtual adjacent-lane car is introduced in the feature extraction part.

The results are shown in Fig. 8.12, where the red, blue and yellow rectangles represent the ego-vehicle, the front car and the virtual car on adjacent lane, respectively. The blue bars are the rear positions of the front car. The corresponding velocity profiles show that with the proposed hierarchical structure, the ego vehicle can decelerate in-time and perform car-following maneuver with the end speed equal to that of the front car. On the other hand, with a short-term MPC, the ego vehicle collides with the front car due to its short preview horizon, even a sharp brake is attempted in the end.

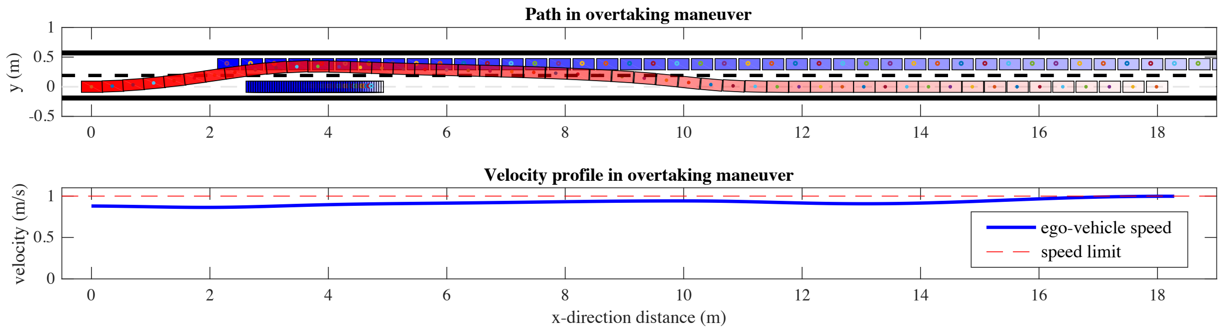


**Figure 8.10:** Overtaking maneuver using the proposed planning and control framework

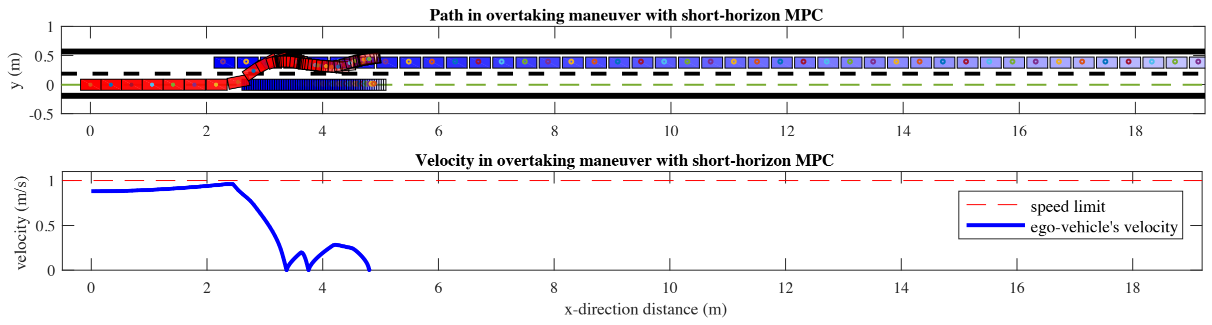


**Figure 8.11:** Overtaking maneuver using only a short-horizon MPC without the learned policy

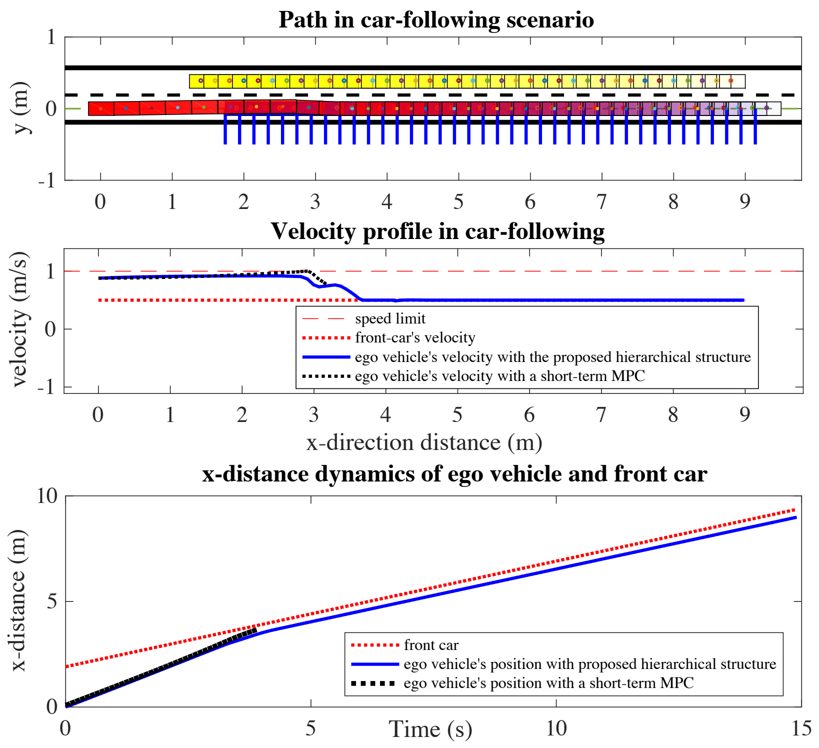


**Figure 8.12:** Car following Performance with the proposed framework

### 8.5.3 Imitation Performance Improvement with DAgger

In this section, the imitation learning performance with DAgger is presented. As mentioned above, the initial size of the training dataset is 20k, and in each iteration 10k on-policy data is generated. To evaluate the imitation performance, we adopt two emergent metrics: the collision rate (the fraction of trajectories where the ego vehicle intersects with another road participate) and the hard brake rate (the frequency at which the ego vehicle brakes harder than $0.9a_{max}^{-}$). Results are shown in Fig. 8.13, where it can seen that both collision and hard brake rates significantly reduce as the DAgger iteration increases. An example of performance improvement via DAgger is also presented in Fig. 8.14, where part (a) represents the on-policy running results with the initial policy $\pi_0$ on training set $D_0$. Due to the data distribution mismatch between the training set and test environment, with $\pi_0$, the policy layer cannot yield correct on-policy reference trajectory. As a result, the ego vehicle fails to perform a smooth lane-keeping maneuver. On the other hand, part (b) shows the results after the DAgger process converges. It can be seen that with more on-policy running data aggregated into the training set D, the performance of the policy layer is effectively improved so that the ego vehicle can smoothly go straight as fast as possible when it is safe.

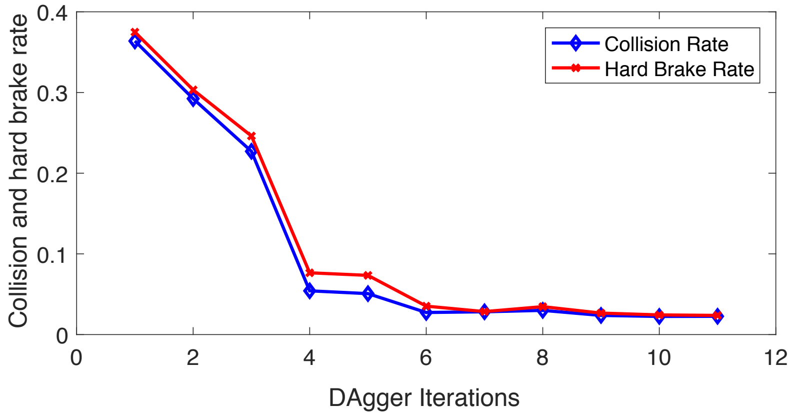


**Figure 8.13:** Improvement of Collision and Hard Brake Rate

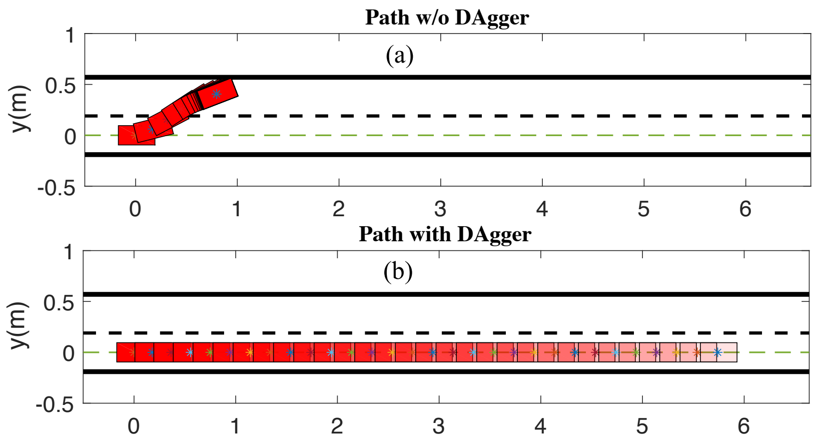


**Figure 8.14:** Learned policy improvement with customized DAgger

### 8.5.4 Comparison in terms of Realtimeness and Optimality

We also compare the online running time for both the proposed hierarchical structure and the baseline expert MPC-based long-term planning policy. With 10 rollouts (i.e., 10k planning events), the worst-case runtime for the expert MPC takes 2.5487s, while the policy layer and the safe execution layer takes $6.42\times10^{-6}$s and 0.0766s, respectively. This means that the proposed hierarchical structure allows us to do planning with a frequency of 10Hz. Even if we try to optimize the problem formulation in the expert MPC policy by constraints approximation, as addressed in [168], a maximum of 2Hz planning is allowed. Therefore, in term of efficiency, the proposed hierarchical structure significantly reduced the runtime so that real-time requirement can be satisfied.

Moreover, we have also compared the performance of the proposed framework with that of the long-term expert MPC policy for trajectories in different homotopy. As shown in Fig. 8.15 and 8.16, five example trajectories are given: lane change, successful and

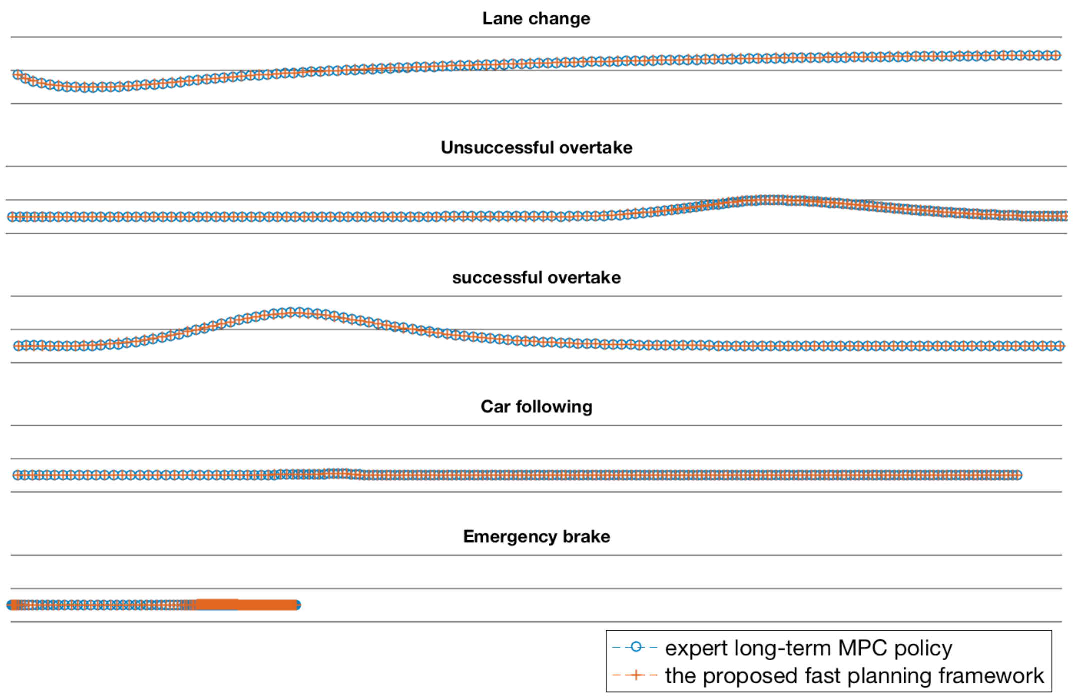


**Figure 8.15:** Performance comparison of the proposed fast planning framework with long-term expert MPC planner on test trajectories with different homotopy

unsuccessful overtake, car following and emergency brake (note that the vehicles are not shown in this plot to make the trajectories clear). We can see that the policy layer and the safe execution layer can achieve similar performance as the long-term MPC policy (the Euclidean distance between the two sets of trajectories are small as shown in Fig. 8.16). The proposed framework, however, is much more efficient for online planning since the the forward computation of policy layer is very fast, and the formulated optimization problem in the execution layer is small due to a short preview horizon.

## 8.6 A Case Study for Real Human Driving

### 8.6.1 The roundabout scenario

We also applied the hierarchical structure to imitate real human driving data on a roundabout scenario. As shown Fig. 8.17, we collect the human driving data and the high-definition map from the "USA_Roundabout_FT" scenario in the INTERACTION dataset [159]. Since the real world driving scenario is much more complicated than the simulated highway driving scenario, we adopted visual features as discussed in Section 8.4.2 in the policy layer and an approximated long-term quadratic programming formulation in the execution layer, as addressed in Section 8.3.1.

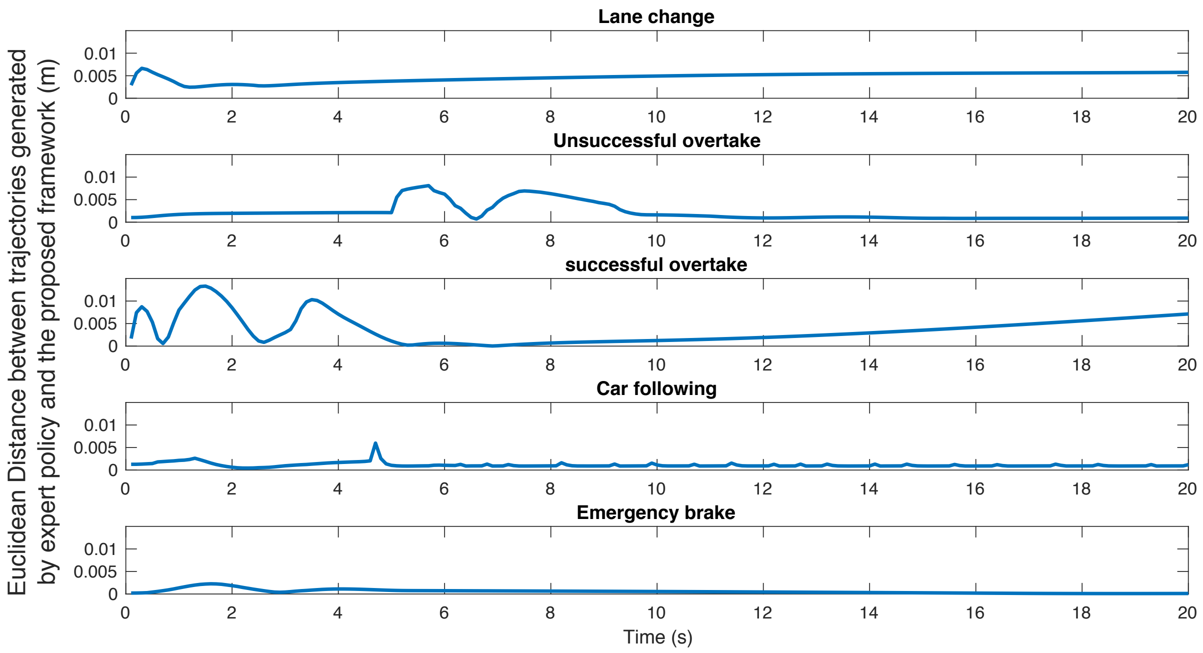


**Figure 8.16:** Euclidean distance between the proposed fast planning framework with long-term expert MPC planner on test trajectories with different homotopy

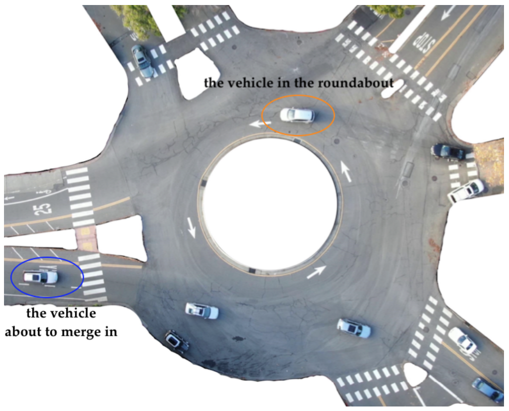


**Figure 8.17:** The roundabout scenario used in imitation learning

## 8.6.2 Results

### The performance of the policy layer

Figure 8.18 shows two examples of the imitation performance of the policy layer. The white rectangle is the ego vehicle and the red ones are the surrounding vehicles. The blue area represents the predicted future trajectory and the green area is the ground truth. We can see that the policy layer can imitate human drivers' behavior quite well.

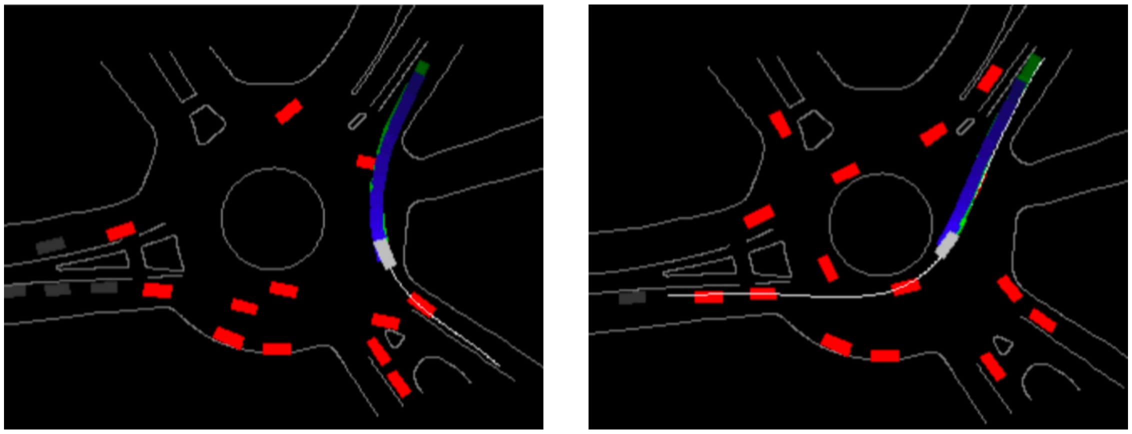


**Figure 8.18:** Two examples of the imitation results of the policy layer: the white rectangle is the ego vehicle and the red ones are the surrounding vehicles. The blue area represents the predicted future trajectory and the green area is the ground truth.

### The performance of the hierarchical structure

In Fig. 8.19 and Fig. 8.20, we show how the proposed hierarchical structure can help enhance the safety and feasibility of the imitation learning performance. In both Fig. 8.19 and Fig. 8.20, the columns with the gray background is with the policy layer only, and the columns with the orange background is with the proposed hierarchical structure. We can see that in Fig. 8.19, at $t = 2.5s, 4.5s, 5.0s$, the hierarchical structure can help avoid collisions with other agents compared to the results using only the policy layer. In Fig. 8.20, the hierarchical structure help to recover the feasibility of the actions generated by the policy layer, particularly for the action from $t = 1.5s$ to $t = 2.0s$.

## 8.7 Chapter Summary

In this chapter, a fast integrated planning and control framework for autonomous driving was proposed based on a combination of imitation learning and optimization. Using a hierarchical structure, the framework can efficiently generate a long-term planning trajectory via imitation, and execute safe and feasible control actions. Moreover, a sampled-DAgger procedure was also introduced to consistently improve the imitation performance. In the policy layer, we explored both numerical features and visual features, and in the execution layers, the performance of two different approximation approaches (a short-term constrained optimization and a long-term approximated quadratic programming) were studied. For verification, a highway driving scenario in simulation and one roundabout scenario with real human driving data were studied with the proposed framework. The results showed that the proposed framework can effectively generate safe, efficient and feasible control actions for real-time applications. Moreover, it showed that with DAgger process, the performance can be consistently

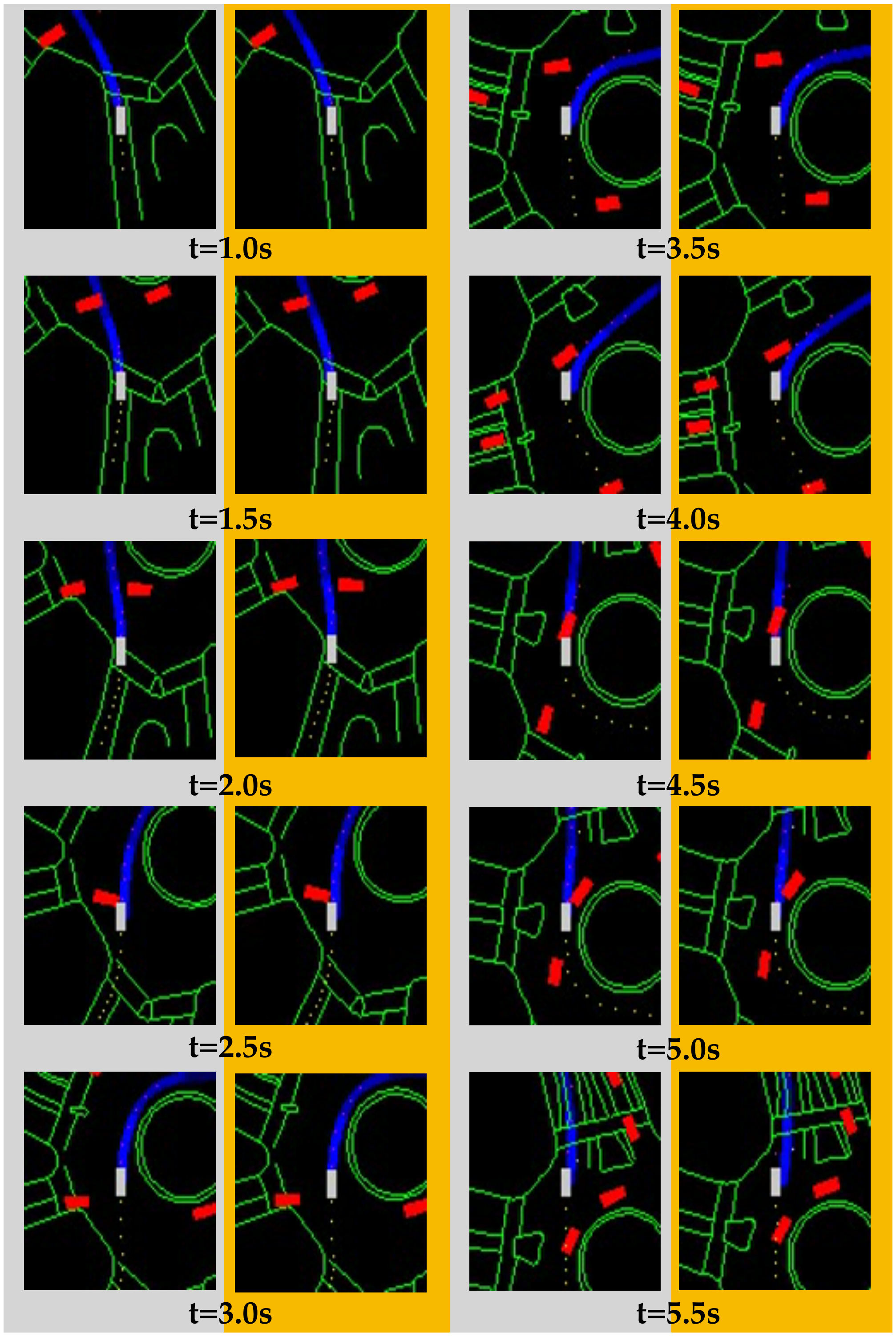


**Figure 8.19:** Performance comparison between the policy layer and the hierarchical structure on collision avoidance

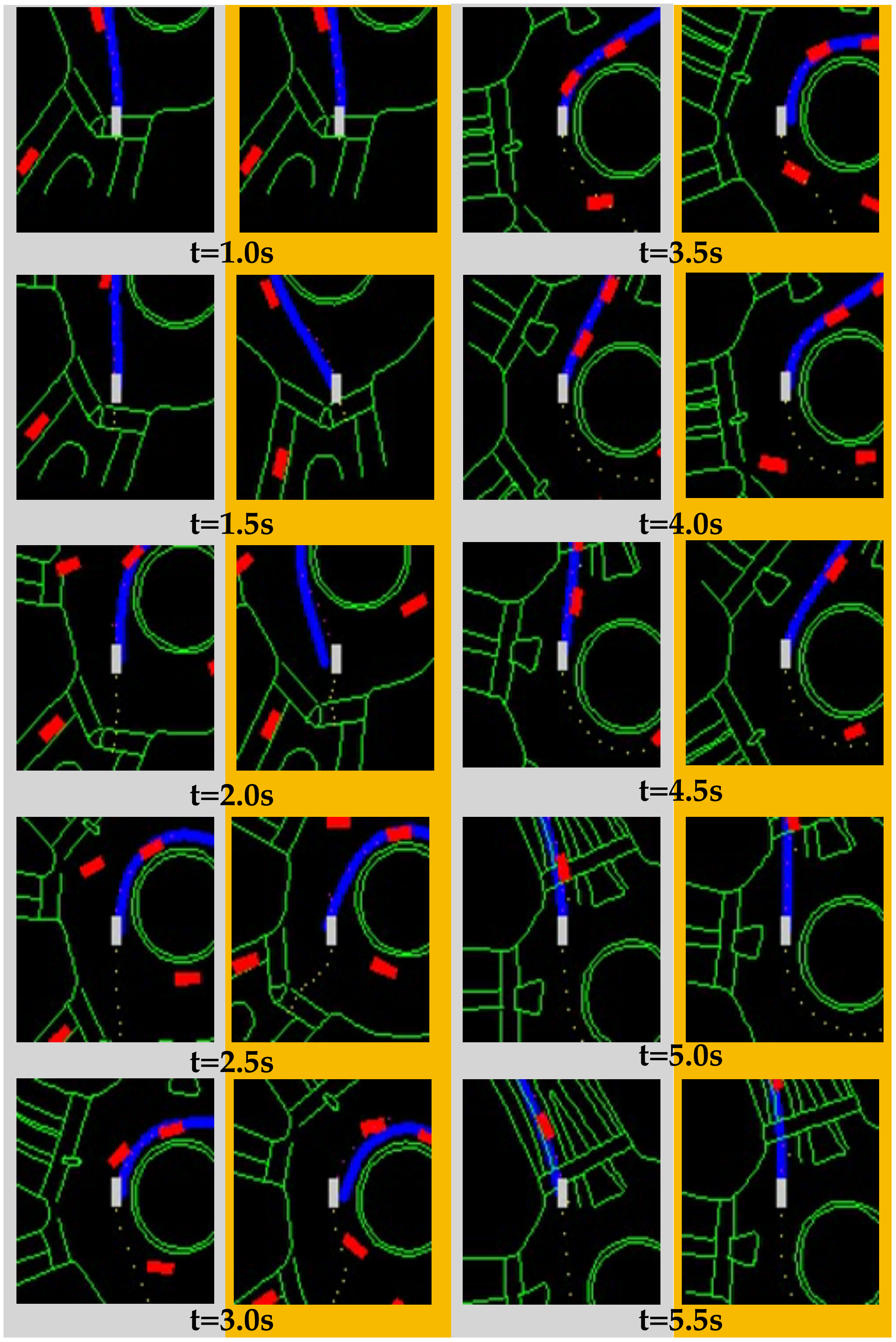


**Figure 8.20:** Performance comparison between the policy layer and the hierarchical structure on feasibility of actions

improved over iterations.

The work in this chapter is a step towards solving complex optimization problems by combining optimization and imitation learning. As autonomous systems are required to handle more and more complicated tasks, we envision such a framework as an effective methodology to address the real-time requirement of the systems.

# Part II

# High-Performance Individual Machine Behavior Design

# Chapter 9

# Multirate Adaptive Disturbance Suppression Beyond Nyquist Frequency

The high-level behaviors designed in Part I need to be executed with high performance to assure safe and efficient interaction between human and autonomous systems. For instance, the maneuver and acceleration commands for autonomous vehicles should be accurately executed via the electronic control unit (ECU). For mobile manipulators, the high-level interactive behaviors will generate low-level commands for the motion motors, unsatisfactory performance of which can directly deteriorate the intelligent behavior and might lead to task failures and even collisions with human. Hence, reliable low-level machine behavior is of equal importance as the high-level intelligent behavior. It is a fundamental support for the high-level hybrid human-machine behavior. In Part II, we will address the problem of high-performance individual machine behavior design in the presence of model uncertainties and external disturbances. Throughout Part II, we are leveraging strength from learning, optimization, and adaptive control to tackle uncertainties and disturbances with different characteristics.

In precision motion systems, suppression of high-frequency unknown disturbances is always critical and challenging. This becomes even more difficult when the system output is only available at limited sampling rates, which is common in digital control, but the frequencies of the disturbances may be beyond the Nyquist frequency of the output samplers. Under such scenarios, the un-availability of the inter-sample outputs may make most of traditional disturbance attenuation techniques useless. To address this problem, a new adaptive disturbance suppression algorithm is proposed in this chapter. Via iterative multirate extended-state estimation and parameter adaptation, the unknown beyond-Nyquist disturbances can be correctly estimated and compensated. Moreover, a special internal model structure is introduced for signals with aliasing frequencies so that the dynamics of the beyond-Nyquist disturbances can be identified more efficiently. We evaluated the performance of the algorithm in simulations on a typical motion systems - hard disk drives, but the general framework can be well applied to other motion systems such as joint motors for manipulators and electronic control units for autonomous vehicles.

## 9.1 Introduction

As shown in Fig. 9.1(a), to attenuate disturbances, typically a digital feedback controller C is designed to regulate the output of the physical system via a holder $\mathcal{H}$ and a sampler $\mathcal{S}$ with limited sampling rates. Based on the frequency properties of disturbances, the controller C needs to generate high gains over the frequency ranges of interest via advanced control techniques such as loop shaping [91], disturbance observers[96], internal model principle (IMP) [35], Youla-parameterization [66], and adaptive controls [26, 124, 128]. Most of these approaches, however, fail if the frequency of the disturbances goes near and even beyond the Nyquist frequency of the sampler. Under these scenarios, the beyond-Nyquist disturbances will excite high-frequency vibrations in the system output that cannot be fully captured by the sampler (as shown in Fig. 9.1(b)), neither for effective feedback regulation nor for unknown parameter identification.

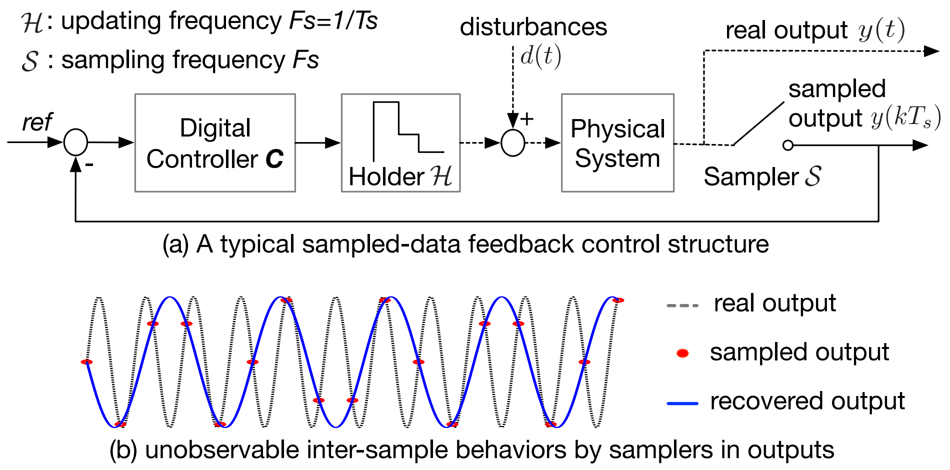

$\mathcal{H}$: updating frequency Fs=1/Ts
$\mathcal{S}$ : sampling frequency Fs
disturbances
$d(t)$
real output $y(t)$
sampled output $y(kT_s)$
ref
-
+
Digital Controller C
Holder $\mathcal{H}$
Physical System
Sampler $\mathcal{S}$
(a) A typical sampled-data feedback control structure
real output
sampled output
recovered output
(b) unobservable inter-sample behaviors by samplers in outputs

**Figure 9.1:** A Rate-limited sampled-data feedback control system and unobservable inter-sample behaviors

To further extend the control effort beyond the Nyquist frequency, extensive algorithms have been proposed along the direction of multirate control. The central idea of multirate control is to update the control signal (i.e., the holder frequency) N times faster than the output sampling frequency so that the inter-sample vibrations can be suppressed or beyond-Nyquist dynamics can be identified, see, for example, multirate repetitive controllers [36, 142], multirate recursive least-square (RLS) parameter identification [100], and multirate reference tracking [151]. Particularly for beyond-Nyquist disturbance attenuation, Atsumi [12] achieved direct beyond-Nyquist loop shaping via multirate resonant filters and frequency analysis of sampled-data systems. This concept is further extended via an add-on forward disturbance-observer structure, for which

Chen et al. [27] introduced a model-based multirate predictor and Yan et al. [152] cast the multirate loop shaping objective into a convex optimization problem. Zheng et al. [164] and Sun et al. [131] proposed to use multirate extended-state observers and lifted-$H_\infty$ approach to achieve simultaneous inter-sample output prediction and disturbance compensation in the presence of system uncertainties. These approaches, however, all assume that the beyond-Nyquist disturbance model is exactly known *a priori*, and explicitly utilize such information for effective disturbance estimation and compensation. Such an assumption, however, does not commonly hold in practice.

In this chapter, *a new multirate control algorithm is proposed to address the remaining challenge with regard to unknown beyond-Nyquist disturbances*. We focus on periodic but unknown disturbance signals whose energy mainly concentrates on several particular frequencies beyond the Nyquist frequency. Compared to previous works, no exact disturbance model is required. Alternatively, a two-step procedure is designed which can automatically update the parameters in the disturbance model so that good estimation can be achieved. Step I: the *disturbance estimation* step, estimates the system states as well as the disturbances via a multirate extended-state observer (MESO) based on slow-rate output samples and the current updated disturbance model. The generated disturbance estimate in Step I is then utilized in Step II, the *model parameter update* step, to further update the disturbance model parameters via recursive least-square (RLS). By iteratively repeating the two steps, the unknown beyond-Nyquist disturbances can be accurately estimated and thus compensated even though their influences to the output cannot be effectively captured by the slow-rate samples. Moreover, the proposed closed-loop system can be reformulated as an add-on multirate observer based compensator that is compatible with any pre-designed feedback controllers so that any existing closed-loop control properties can be well preserved.

## 9.2 Problem formulation

### 9.2.1 Fundamental Limitation for Single-Rate Control

**Proposition 1.** *Consider the sampled-data framework in Fig.9.1(a) with a zero-order hold (ZOH) $\mathcal{H}$ and a sampler $\mathcal{S}$, both working at a frequency of $F_s$. If the disturbance signal $d(t)$ contains frequency components $e^{j\omega_d t}$ beyond the Nyquist frequency of the sampler $\mathcal{S}$, i.e., $\omega_s > \omega_d > \frac{\omega_s}{2}=\pi F_s$, there is no such a single-rate digital controller $C(z)$ with $z = e^{j\omega T_s}$ that can effectively suppress the influence of $d(t)$.*

*Proof.* Suppose that the dynamics of the system can be represented by $P_c(s)$ and its $T_s$-ZOH discrete-time equivalent (pulse transfer function) is $P_d(z)$, $z = e^{j\omega T_s}$. Then with any stabilizing single-rate feedback controller $C(z)$, the relationship between $d(t)$ and the closed-loop system output $y(t)$ can be expressed as (see [131] for details):

$$y(t) = [P_c(j\omega_d) - \mathcal{T}(\omega_d, 0)]\, e^{j\omega_d t} - \sum_{l=-\infty, l\neq 0}^{+\infty} \mathcal{T}(\omega_d, l) e^{(j\omega_d + j\omega_s l)t} \tag{9.1}$$

where $\mathcal{T}(\omega,l) = \frac{1}{T_s}\frac{P_d(e^{j\omega T_s})C(e^{j\omega T_s})\mathcal{H}(j\omega+j\omega_s l)P_c(j\omega+j\omega_s l)}{1+P_d(e^{j\omega T_s})C(e^{j\omega T_s})}$. Noting that when $\omega_d > \dfrac{\omega_s}{2}$, we have $\mathcal{H}(j\omega_d) < \mathcal{H}(j\omega_d - j\omega_s) \approx T_s$, which makes the tailing term with $l = -1$ in (9.1) innegligible. Moreover, the first term $P_c(j\omega_d) - \mathcal{T}(\omega_d, 0)$ can be simplified as

$$\left[1 - \frac{1}{T_s}\frac{P_d(e^{j\omega_d T_s})C(e^{j\omega_d T_s})\mathcal{H}(j\omega_d)}{1+P_d(e^{j\omega_d T_s})C(e^{j\omega_d T_s})}\right] P_c(j\omega_d),$$

the gain of which at $\omega_d$ decreases as $|C(e^{j\omega_d T_s})|$ increases. The tailing term with $l = -1$, however, becomes

$$\begin{aligned} \mathcal{T}(\omega_d, -1) &= \frac{1}{T_s}\frac{P_d(e^{j\omega_d T_s})C(e^{j\omega_d T_s})\mathcal{H}(j\omega_d - j\omega_s)P_c(j\omega_d - j\omega_s)}{1+P_d(e^{j\omega_d T_s})C(e^{j\omega_d T_s})} \\ &\approx \frac{P_d(e^{j\omega_d T_s})C(e^{j\omega_d T_s})P_c(j\omega_d - j\omega_s)}{1+P_d(e^{j\omega_d T_s})C(e^{j\omega_d T_s})}, \end{aligned} \tag{9.2}$$

the gain of which increases as $|C(e^{j\omega_d T_s})|$ increases. This means that with a single-rate $C(z)$, it is impossible to simultaneously reduce the gains of the first term and the tailing term in (9.1). Hence, the disturbance $d(t)$ will generate aliasing phenomenon in $y(t)$ that cannot be eliminated by a single-rate $C(z)$. □

## 9.2.2 Multirate Observer based Compensation

Given the fundamental limitation from Proposition 1, a multirate Youla-parameterized control structure, as shown in Fig. 9.2, has been proposed to compensate for the beyond-Nyquist disturbances in our previous work [131]. In Fig. 9.2, the output is available at a samping period of $T_s = NT_f$, $N \geqslant 2$. On top of an existing slow-rate controller $C(z_{NT_f})$, an add-on multirate extended-state observer (MESO) is introduced to generate a fast compensation signal $c(kT_f)$. As shown in Fig. 9.2, $c(kT_f)$ is nothing but the estimate of the disturbance. $\mathcal{H}_{NT_f}$ and $\mathcal{H}_{T_f}$, respectively, represent the discrete-time ZOH and D/A ZOH working at sampling periods of $NT_f$ and $T_f$.

Assume that the continuous-time beyond-Nyquist disturbance signal $d(t)$ can be well represented by the output of a fast discrete system $G_D(z)(z = e^{j\omega T_f})$ driven by white noises. Particularly for periodic disturbance signals, if $d(t)$ contains a frequency peak at $\omega_d$ satisfying $\frac{\omega_s}{2} < \omega_d < \omega_s$, then the disturbance dynamics $G_D$ can be approximated by a narrow-band bandpass filter. Hence, the frequency-domain and controllable state-space representations of $G_D$ can, respectively, be expressed as

$$\begin{aligned} G_D(z) &= 1-\frac{1-2\cos(\omega_d T_f)z^{-1}+z^{-2}}{1-2\alpha\cos(\omega_d T_f)z^{-1}+\alpha^2 z^{-2}}, \alpha \in [0.98, 0.999] \\ &= \frac{2(1-\alpha)\cos(\omega_d T_f)z^{-1}+(\alpha^2-1)z^{-2}}{1-2\alpha\cos(\omega_d T_f)z^{-1}+\alpha^2 z^{-2}}. \end{aligned} \tag{9.3}$$

$$G_D : \left\{ \begin{aligned} x_d(k+1) &= A_d x_d(k)+B_d n(k), \\ d(k) &= C_d x_d(k), \end{aligned} \right. \tag{9.4}$$

where $\alpha$ is a hyper-parameter tuning the width of the passband of $G_D$ and $n(k)$ is the white noise driving the disturbance model.

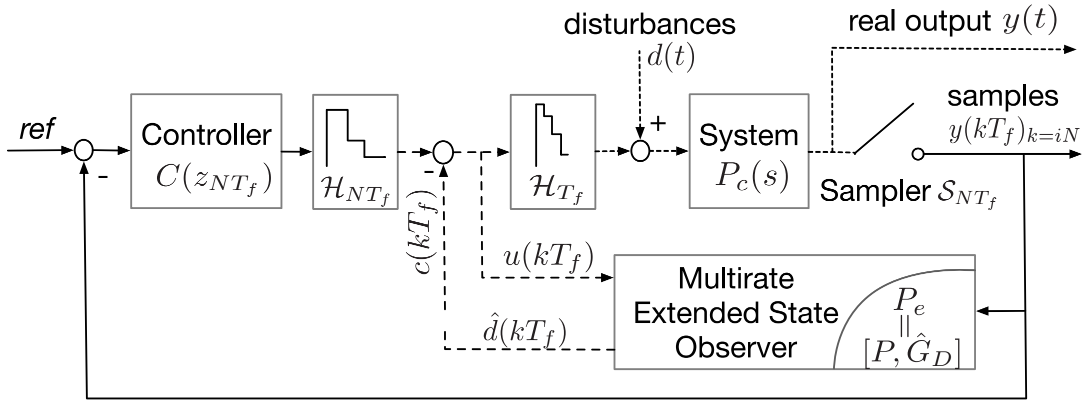


**Figure 9.2:** The multirate observer based compensation

Similarly, a $T_f$-sampled ZOH equivalent of the system dynamics $P_c(s)$ can be obtained as

$$P: \left\{ \begin{aligned} x_p(k+1) &= A_p x_p(k)+B_p u(k)+B_p d(k), \\ y(k) &= C_p x_p(k)+v(k) \end{aligned} \right. \tag{9.5}$$

Note that $v(k)$ is the measurement noise and assumed to be white and uncorrelated with $n(k)$ in (9.4). Defining new system states as $x_e(k)=[x_p(k);x_d(k)]$ and substituting (9.4) into (9.5), we can obtain an augmented system $P_e(z)$:

$$\begin{aligned} x_e(k+1) &= \underbrace{\left[\begin{array}{c:c} A_p & B_p C_d \\ \hdashline \mathbf{0}_{n_d\times n_p} & A_d \end{array}\right]}_{A_e} x_e(k)+\underbrace{\left[\begin{array}{c} B_p \\ \hdashline \mathbf{0}_{n_d\times 1} \end{array}\right]}_{B_{ue}} u(k)+\underbrace{\left[\begin{array}{c} \mathbf{0}_{n_p\times 1} \\ \hdashline B_d \end{array}\right]}_{B_{ne}} n(k), \\ y(k) &= \underbrace{\left[\begin{array}{c:c} C_p & \mathbf{0}_{1\times n_p} \end{array}\right]}_{C_{ye}} x_e(k)+v(k), \\ d(k) &= \underbrace{\left[\begin{array}{c:c} \mathbf{0}_{1\times n_d} & C_d \end{array}\right]}_{C_{de}} x_e(k). \end{aligned} \tag{9.6}$$

Based on this augmented system and the fact that only $y(k)$ at $k{=}iN, i{=}0,1,\cdots$ are available, a multirate Kalman Filter[127] can be designed which predicts at the fast rate ($T_f$) and corrects at the slow rate ($T_s{=}NT_f$), as given in (9.7) and (9.8). Note that in (9.7) and (9.8), $\hat{A}_e{=}\left[\begin{array}{c|c} A_p & B_p\hat{C}_d \\ \hline \mathbf{0}_{n_d\times n_p} & \hat{A}_d \end{array}\right]$ and $\hat{C}_{de}{=}\left[\begin{array}{c|c} \mathbf{0}_{1\times n_d} & \hat{C}_d \end{array}\right]$ represent the augmented system matrices with the assumed disturbance model $\hat{G}_D{=}\left[\begin{array}{c|c} \hat{A}_d & B_d \\ \hline \hat{C}_d & 0 \end{array}\right]$ (the uncertainties in disturbance dynamics $G_D$ will not influence $B_d$ due to the controllable state-space realization we choose in (9.4)).

$$\text{Prediction}:\begin{cases} \hat{x}_e(k{+}1|k)=\hat{A}_e\hat{x}_e(k){+}B_{ue}u(k), \\ \hat{y}(k)=C_{ye}\hat{x}_e(k), \qquad \hat{d}(k)=\hat{C}_{de}\hat{x}_e(k), \\ M(k{+}1)=\hat{A}_eZ(k)\hat{A}_e^T+B_{ne}WB_{ne}^T \end{cases} \tag{9.7}$$

$$\text{Correction}:\begin{cases} \hat{x}_e(k{+}1|k{+}1) &= \hat{x}_e(k{+}1|k){+}F(k{+}1) \\ & \quad \times\big(y(k){-}C_{ye}\hat{x}_e(k{+}1)\big), k{=}iN \\ \hat{x}_e(k{+}1|k{+}1) &= \hat{x}_e(k{+}1|k), k\neq iN \\ F(k{+}1) &= M(k{+}1)C_{ye}^T \\ & \quad \times\left(C_{ye}M(k{+}1)C_{ye}^T{+}V\right)^{-1} \\ Z(k{+}1) &= \big(I{-}F(k{+}1)C_{ye}\big)M(k{+}1), k=iN \\ Z(k{+}1) &= M(k{+}1), k\neq iN \end{cases} \tag{9.8}$$

*Remark 1*: In this work, our emphasis is on uncertainties of the beyond-Nyquist disturbance model and assume that the system model is precisely known, particularly over the frequency ranges where the beyond-Nyquist disturbance occurs. Such precise system models are necessary as explained in Proposition 2 below and can be obtained via multirate adaptive identification methods, e.g., [100].

**Proposition 2.** *With the multirate Kalman filtering defined in (9.7) and (9.8), if $P_e(z)$ in (9.6) is stable, and $(A_e, B_{ne})$ and $(A_e, C_{ye})$ are, respectively, controllable and observable, then:*

1. *the control structure given in Fig. 9.2 is a Youla-parameterized multirate stabilizing feedback controller for any pre-existing stabilizing slow-rate controller $C$. Moreover, optimal disturbance estimation (in the sense of $L_2$) and effective compensation can be guaranteed if the disturbance model is exactly known a priori, i.e., $\hat{G}_D{=}G_D$.*

2. *If $\hat{G}_D\neq G_D$, the disturbance estimate $\hat{d}(kT_f)$ via the MESO structure in Fig. 9.2 and (9.7) and (9.8) will be biased. Moreover, $\hat{d}(kT_f)$ contains aliased frequency components of $d(t)$. As a consequence, direct cancellation of $d(t)$ by $\hat{d}(kT_f)$ will fail and may cause closed-loop instability.*

*Proof.* ***Part 1***: This can be directly concluded from previous work [131] and thus is omitted. One can refer [131] for details.

***Part 2***: First, the biased estimation of disturbance is straightforward in this case. When $\hat{G}_D(z) \neq G_D(z)$, the forward dynamics used in the observer in (9.7) from $n(k)$ to $y(k)$ can be reformulated as $y(k) = P(q^{-1})G_D(q^{-1})n(k) = P(q^{-1})\hat{G}_D(q^{-1})\left[\frac{G_D(q^{-1})}{\hat{G}_D(q^{-1})}n(k)\right]$ ($q^{-1}$ is the one-step delay operator defined by $x(k) = q^{-1}x(k+1)$). This indicates that the equivalent process noise $\left[\frac{G_D(q^{-1})}{\hat{G}_D(q^{-1})}n(k)\right]$ in the Kalman Filter model is not white but coloured. Therefore, the disturbance estimate will be biased.

Second, to show that the disturbance estimate $\hat{d}(kT_f)$ contains aliased frequency components of $d(t)$, we first consider the estimator, i.e., setting $c(kT_f)$=0 in Fig. 9.2. In steady state, the multirate Kalman filtering process in (9.7) and (9.8) yields a 1-input-N-output slow-rate ($T_s$) time-invariant relationship from $y(kN)$ to $\hat{\mathbf{d}}(i)$=$[d(iNT_f), d((iN+1)T_f), \cdots, d((iN+N-1)T_f)]^T$, i.e., $\hat{\mathbf{d}}(i) = \left[G_1(q^{-N}), G_2(q^{-N}), \cdots, G_N(q^{-N})\right]^T y(iN)$ (See [131] for details). Hence, the spectrum of $\hat{d}(kT_f)$ can be expressed as

$$\begin{aligned}
\hat{D}(e^{j\frac{\omega T_s}{N}}) &= \sum_{k=0}^{\infty}\hat{d}(k)e^{\frac{-jk\omega T_s}{N}} = \sum_{i=0}^{\infty}\sum_{m=1}^{N-1}\hat{d}(iN+m)e^{\frac{-j(iN+m)\omega T_s}{N}} \qquad (9.9)\\
&= \sum_{i=0}^{\infty}\sum_{m=1}^{N-1}G_m(e^{\frac{jN\omega T_s}{N}})e^{\frac{-jm\omega T_s}{N}}y(iN)e^{\frac{-j(iN)\omega T_s}{N}}\\
&= \sum_{m=1}^{N-1}G_m(e^{j\omega T_s})e^{\frac{-jm\omega T_s}{N}}\sum_{i=0}^{\infty}y(iN)e^{-ji\omega T_s}\\
&= \left[\sum_{m=1}^{N-1}G_m(e^{j\omega T_s})e^{\frac{-jm\omega T_s}{N}}\right]Y(e^{j\omega T_s}).
\end{aligned}$$

where $Y(e^{j\omega T_s})$ is the spectrum of the slow-sampled sequence $y(iNT_f)$ and satisfies $Y(e^{j\omega T_s})=\frac{1}{T_s}\sum_{l=-\infty}^{\infty}Y_t(j(\omega+l\omega_s))$ where $Y_t(j\omega)$ represents the spectrum of $y(t)$. Recall the fact from (9.1) that, with a slow-rate feedback controller C, the system $y(t)$ contains both the real and aliased frequency components at $\omega_d$ and $\omega_s-\omega_d$ if $d(t) = e^{j\omega_d t}$. Moreover, note that all $G_m(e^{j\omega T_s}), m=1,2,\cdots$ are periodic with a period of $\omega_s$, i.e., $G_m(e^{j\omega T_s})=G_m(e^{j(\omega+l\omega_s)T_s}), l \in \mathcal{N}$. Therefore, the magnitude of $\left[\sum_{m=1}^{N-1}G_m(e^{j\omega T_s})e^{\frac{-jm\omega T_s}{N}}\right]$ is also periodic, and thus the spectrum of $\hat{d}(kT_f)$ periodically contains frequency components at the true and the aliased frequencies $\omega_d$, $\omega_s-\omega_d$, $\omega_d+\omega_s$, $2\omega_s-\omega_d, \cdots$, $\omega_s l+\omega_d$, $\omega_s(l+1)-\omega_d, \cdots$. □

*Remark 2*: Note that the above analysis holds for every frequency $\omega_d$ in the disturbance signal. If the disturbance contains multiple frequency components, e.g., $\omega_{d_1}, \omega_{d_2}, \cdots, \omega_{d_n}$, then the estimate signal $\hat{d}(kT_f)$ contains multiple periodic signals with frequencies at $\omega_{d_i}$, $\omega_s-\omega_{d_i}$, $\omega_{d_i}+\omega_s$, $2\omega_s-\omega_{d_i}, \cdots$, $\omega_s l+\omega_{d_i}$, $\omega_s(l+1)-\omega_{d_i}, \cdots$ with $i = 1, 2, \cdots, n$.

## 9.3 Adaptive Multirate Extended-State Observer

To accurately estimate the unknown beyond-Nyquist disturbance, an adaptive multirate observer is proposed, as shown in Fig. 9.3. Compared to previous work in Fig. 9.2, a parameter adaptation step based on RLS is introduced in addition to the MESO so that the disturbance model $\hat{G}_D$ used in (9.7) and (9.8) can be further updated to achieve optimal disturbance estimation and effective compensation.

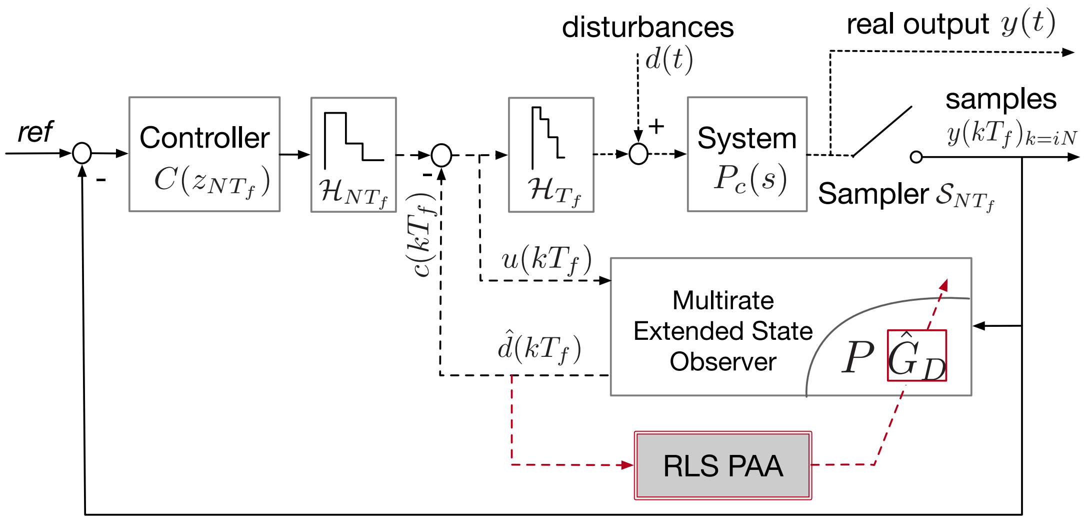


**Figure 9.3:** The proposed adaptive multirate Observer Based Compensation

### 9.3.1 Disturbance Model Update via Parameter Adaptation

As discussed in Section 2, if $d(t)$ is a periodic signal which contains $n$ beyond-Nyquist frequency peaks at $\omega_{d_i}, i = 1, 2, \cdots, n$ and the assumed model $\hat{G}_D \neq G_D$, the steady-state disturbance estimate $\hat{d}(kT_f)$ will generate periodic frequency peaks at $\omega_s l+\omega_{d_i}$ and $\omega_s(l+1)-\omega_{d_i}$, $l=0,1,\cdots$. Recall that for sequences $b(kT_f) = e^{jk\omega T_f}, \omega < \frac{\pi}{T_f}$, $k=0,1,\cdots$, $(1-2\cos(\omega T_f)q^{-1}+q^{-2})b(kT_f) = 0$ holds with $q^{-1}b((k+1)T_f)=b(kT_f)$. Therefore, for signal $\hat{d}(kT_f)$, with even $N$, we have

$$\prod_{i=1,2,\cdots,n} \prod_{l=0,1,\cdots}^{(\omega_s l+\omega_{d_i})<\frac{\pi}{T_f}} \left[1-2\cos((\omega_s l+\omega_{d_i})T_f)q^{-1}+q^{-2}\right] \times \left[1-2\cos((\omega_s(l+1)-\omega_{d_i})T_f)q^{-1}+q^{-2}\right] \hat{d}(kT_f) = 0. \tag{9.10}$$

Note that with $T_f=T_s/N$, we have $l=0,1,\cdots,\frac{N}{2}-1$. Let $\cos(\omega_{d_i}T_f) \triangleq \theta_i$ and $\sin(\omega_{d_i}T_f) \triangleq \sqrt{1-\theta_i^2}$, then all $\frac{N}{2}$ pairs of $\cos((\omega_s l+\omega_{d_i})T_f)$ and $\cos((\omega_s(l+1)-\omega_{d_i})T_f)$ can be reorganized and expressed as

$$\begin{aligned}\cos((\omega_s l+\omega_{d_i})T_f) &= \cos(\frac{2\pi l}{N})\theta_i-\sin(\frac{2\pi l}{N})\sqrt{1-\theta_i^2},\\ \cos((\omega_s(l'+1)-\omega_{d_i})T_f)|_{l'=N/2-1-l} &= \cos((\omega_s(N/2-l)-\omega_{d_i})T_f)\\ &= \cos(\pi-(\omega_s l+\omega_{d_i})T_f)\\ &= -\cos((\omega_s l+\omega_{d_i})T_f).\end{aligned}$$

As shown in Fig. 9.4, for $N$=2, 4, we have

- $N$=2: $l=0 \Rightarrow \cos(\omega_{d_i}T_f) = \theta_i$, $\cos(\pi-\omega_{d_i}T_f) = -\theta_i$, and the disturbance model in (9.10) becomes $(1-2\theta_i q^{-1} + q^{-2})(1 + 2\theta_i q^{-1}+q^{-2}) = \left[1 + (2-4\theta_i^2)q^{-2} + q^{-4}\right] =0$.

- $N$=4: $l=0 \Rightarrow \cos(\omega_{d_i}T_f)=\theta_i$ and $\cos(\pi-\omega_{d_i}T_f)=-\theta_i$;
  $l=1 \Rightarrow \cos((\omega_{d_i} + \omega_s)T_f) = \cos(\omega_{d_i}T_f + \pi/4) = \sqrt{1-\theta_i^2}$,
  $\cos(\pi-(\omega_{d_i}+\omega_s)T_f)=-\sqrt{1-\theta_i^2}$, which simplifies the disturbance model in (9.10) as $\left[1+(2-4\theta_i^2)q^{-2}+q^{-4}\right] \times \left[1-(2-4\theta_i^2)q^{-2}+q^{-4}\right] = \left[1+(2-(2-4\theta_i^2)^2)q^{-4}+q^{-8}\right]$.

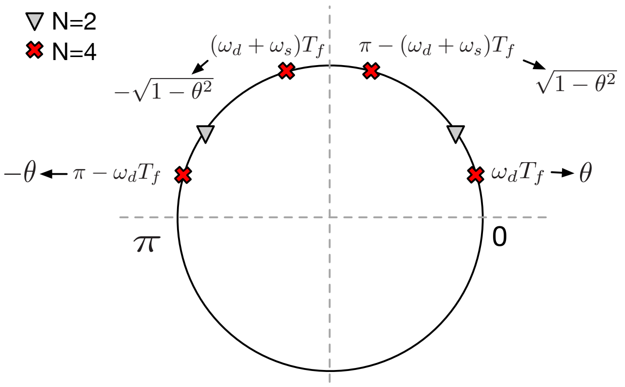

▽ N=2
✖ N=4
$(\omega_d+\omega_s)T_f$
$\pi-(\omega_d+\omega_s)T_f$
$-\sqrt{1-\theta^2}$
$\sqrt{1-\theta^2}$
$-\theta \leftarrow \pi-\omega_d T_f$
$\omega_d T_f \rightarrow \theta$
$\pi$
0

**Figure 9.4:** Distribution of zeros with different N for base frequency at $\omega_d$

Moreover, as shown in Fig. 9.4, for larger N, as long as the symmetry holds, i.e., $N=2^p, p\in\mathcal{N}_+$, (9.10) can always be simplified in a "squared" format given by (9.11) where the unknown parameter $\theta_i=\cos(\omega_{d_i}T_f)$ only influence the second term, denoted by $f(\theta_i)$. For instance, as shown above, for $N = 2$ and $N = 4$, we have $f(\theta_i)=2-4\theta_i^2$ and $f(\theta_i)=2-(2-4\theta_i^2)^2$, respectively.

$$\prod_{i=1,2,\cdots,n}\left[1-f(\theta_i)q^{-N}+q^{-2N}\right]\hat{d}(kT_f) = 0 \tag{9.11}$$

Further expanding (9.11), we get

$$\begin{aligned}
&\prod_{i=1,2,\cdots,n}\left[1-f(\theta_i)q^{-N}+q^{-2N}\right]\\
\triangleq\;& 1+w_1q^{-N}+w_2q^{-2N}+\cdots+w_nq^{-nN}+\cdots+w_1q^{-2(n-1)N}+q^{-2nN}\\
=\;& 1+q^{-2nN}+\sum_{m=1}^{n-1}w_m(q^{-mN}+q^{-(2n-m)N})+w_nq^{-nN}, \qquad (9.12)
\end{aligned}$$

where $w_m, m=1,2,\cdots,n$ are functions of $f(\theta_i), i=1,2,\cdots,n$ ($f(\theta_i)$ is defined in Eq. 9.11). Hence, we have

$$\hat{d}(kT_f)+\hat{d}((k-2nN)T_f)=-\mathbf{w}^T\Phi(k), \qquad (9.13)$$

where $\mathbf{w}=[w_1,w_2,\cdots,w_n]^T$ is the unknown parameter vector and $\Phi(k)\triangleq[q^{-N}+q^{-2(n-1)N}, \cdots, q^{-(n-1)N}+q^{-(n+1)N}, q^{-nN}]^T\hat{d}(kT_f)$. Based on (9.13), the recursive least-square (RLS) parameter adaptation algorithm (PAA) can be utilized to estimate $\mathbf{w}$. Then $\hat{\mathbf{w}}$ can be solved for $f(\theta_i)$s and $\theta_i$s to give update for $\hat{G}_D$ via the relationship in (9.3) and (9.4). Define $\hat{d}(kT_f)+\hat{d}((k-2nN)T_f)\triangleq b(k)$, and the parameter estimate $\hat{\mathbf{w}}(k)$ in PAA is updated as follows:

$$\hat{\mathbf{w}}(k)=\hat{\mathbf{w}}(k-1)+F(k-1)\Phi(k)\frac{\epsilon^0(k)}{1+\Phi^T(k)F(k-1)\Phi(k)} \qquad (9.14)$$

$$\epsilon^0(k)=b(k)-\hat{b}(k)=b(k)-\hat{\mathbf{w}}^T(k-1)\Phi(k) \qquad (9.15)$$

$$F(k)=\frac{1}{\lambda(k-1)}\left\{F(k-1)-\frac{F(k-1)\Phi(k)\Phi^T(k)F(k-1)}{\lambda(k-1)+\Phi^T(k)F(k-1)\Phi(k)}\right\} \qquad (9.16)$$

An exponential forgetting factor $\lambda(k)=\lambda_{end}-[\lambda_{end}-\lambda(k-1)]\lambda_0$ is used to improve the convergence speed [83].

## 9.3.2 Iterative Multirate Estimation and Adaptation

As shown in Fig. 9.3, to accurately estimate the beyond-Nyquist periodic disturbances, the multirate estimation step given in (9.7) and (9.8) and the parameter adaptation step in (9.14) and (9.16) are iteratively processed. The algorithm is summarized below:

1. *Initialization*: Initial guess of $\hat{G}_D^0$ and configuration of all initial parameters in the multirate Kalman Filter and PAA;

2. *Estimation Step*: For j-th iteration, with disturbance model $\hat{G}_D^{j-1}$, system measurements $y(kT_f)|_{k=iN}$ and (9.7-9.8), obtain a sequence of $\hat{d}^j(kT_f)$;

3. *Parameter Update Step*: With $\hat{d}^j(kT_f)$, run PAA in (9.14-9.16) to update $\hat{G}_D^j$;

4. Repeat step 2-3 until the adaptation process converges, i.e., $|\omega_{d_i}^{j} - \omega_{d_i}^{j-1}| < \epsilon$ with a pre-defined threshold $\epsilon$ for all $i = 1, 2, \cdots, n$.

*Remark 3*: For each iteration, multiple steps of estimation can be run before parameter update step and vice versa. Moreover, for scenarios where the frequency characteristics of the beyond-Nyquist disturbances do not change, the algorithm can be simplified into three sequential steps instead of being iterative, namely: 1) *a priori* disturbance estimation for $\hat{d}(kT_f)$; 2) identification of $\omega_d$ via $\hat{d}(kT_f)$; and 3) *a posteriori* disturbance estimation $\hat{d}(kT_f)$ with $\omega_{d_i}$.

### 9.3.3 Convergence of the Iterative Process

As discussed in Proposition 2, the disturbance estimate $\hat{d}(kT_f)$ contains frequency components that satisfy the internal model in (9.11) as long as the MESO is stable and the assumed disturbance model mismatches the true disturbance model. Therefore, in the iterative process, the parameter update step is assured to converge to $\omega_d$s. Once it converges, using Part 1 of Proposition 2, it is guaranteed that unbiased disturbance estimate can be obtained for effective compensation.

## 9.4 Case Study

### 9.4.1 The Hard Disk Drive System

A single-stage HDD system in track-following mode is utilized for simulations in this section. In such HDD systems, the read/write arm is actuated by a voice coil motor (VCM) and our goal is to regulate the arm to follow the tracks in the presence of various disturbances. As a sampled-data control system, the position error signal (PES) is available only at a limited sampling rate ($T_s$) for feedback regulation. With a typical configuration of $T_s$=3.7879×$10^{-5}$s, a slow-rate stabilizing controller C can be designed for a closed-loop bandwidth of around 800Hz.

Figure 9.5 shows the frequency responses of a benchmark HDD model [43], including both the continuous-time model and two ZOH-based discrete-time approximations with $T_s$ and $T_f = T_s/N, N=4$, respectively. It is clear that beyond the Nyquist frequency of the system $F_N = F_s/2 = 1/(2T_s)$, there is a resonance in the continuous-time HDD dynamics. In the following simulations, we assume that the disturbance entering the HDD system has a frequency peak around the beyond-Nyquist resonance ($f_d$=16.6kHz) so that the resonance will be excited to generate unobservable high-frequency vibrations in the system output.

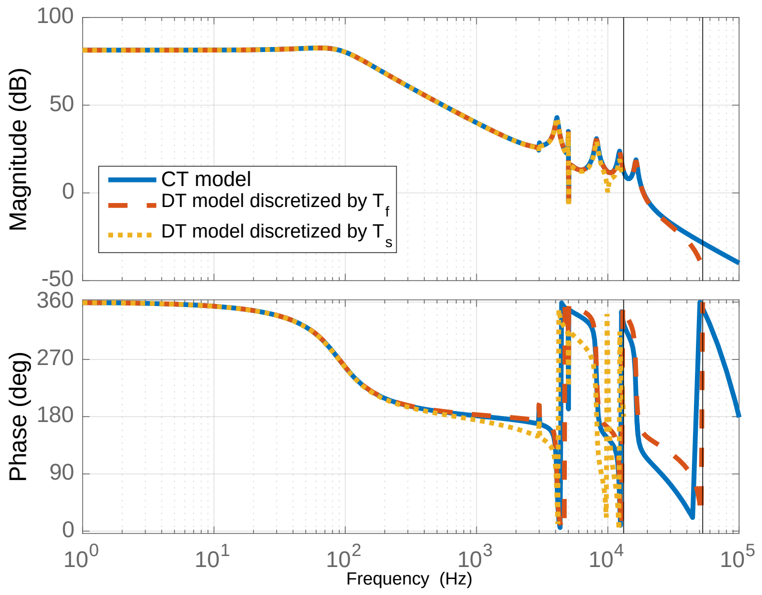


**Figure 9.5:** Frequency Responses of the HDD model

## 9.4.2 Simulation Results

### Biased disturbance estimate with model mismatches

Figure 9.6 and Fig. 9.7 present the estimates of the system output $y(kT_f)$ and disturbance $d(kT_f)$ with accurate ($\hat{G}_D{=}G_D$) and inaccurate ($\hat{G}_D{\neq}G_D$) disturbance models, respectively. Clearly, one can see that with accurate beyond-Nyquist disturbance model, the multirate extended-state observer based estimation given in (9.7) and (9.8) can accurately recover the unobservable inter-sample behaviours in $y(t)$ and yield unbiased disturbance estimate, as addressed in part 1 of Proposition 2. On the contrary, if $\hat{G}_D{\neq}G_D$, then as shown in Fig. 9.7, neither the inter-sample behaviors in $y(t)$ nor the disturbance estimate match the real ones. Moreover, Figure 9.8 provides the spectra of $\hat{d}(kT_f)$ with accurate and inaccurate $G_D$. It is shown that when $\hat{G}_D{\neq}G_D$, $\hat{d}(kT_f)$ will contain both the true frequency components $f_d$ as well as its aliasing counterpart $F_s{-}F_d{=}9.8$kHz and at higher frequencies with a frequency period of $F_s{=}1/T_s{=}26400$Hz.

### Performance improvement via two-step estimation and PAA

Figure 9.9 shows the results of the proposed two-step procedure. Initially, with an inaccurate guess of $G_D$, the estimation of both $y(t)$ and $d(t)$ is poor. The performance, however, is significantly improved via the parameter identification process given in (9.14) and (9.16). Both the system output and the unknown beyond-Nyquist disturbances are correctly estimated. Such results can also be verified by the parameter convergence profile given in Fig. 9.10, where one can clearly see that the parameters as well as the disturbance dynamics model converge to the real ones.

**Closed-loop compensation performance**

Once the parameter converges, the disturbance estimate with the identified $\hat{G}_D$ can then be used to close the internal loop in Fig. 9.3 as a compensation signal, i.e., $c(kT_f) = \hat{d}(kT_f)$. Figure 9.11 shows the closed-loop compensation performance in time domain, in which the influences of the beyond-Nyquist disturbance has been effectively suppressed.

## 9.5 Chapter Summary

In this chapter, a beyond-Nyquist disturbance suppression control scheme was proposed based on a combined process with multirate extended-state estimation and parameter adaptation. Compared to previous works that require accurate prior knowledge about the disturbance dynamics, the proposed approach can directly identify the unknown disturbance dynamics and utilize the identified model for unbiased output and disturbance estimation. Simulation results on a HDD benchmark track-following problem were provided, with a periodic beyond-Nyquist disturbance. It was shown that the identified disturbance model converges to the real one, and both the inter-sample behaviors in the system output and the high-frequency disturbance are accurately estimated. Moreover, the closed-loop simulation results showed the proposed control structure, as a Youla-parameterized multirate controller, can effectively compensate the beyond-Nyquist disturbance.

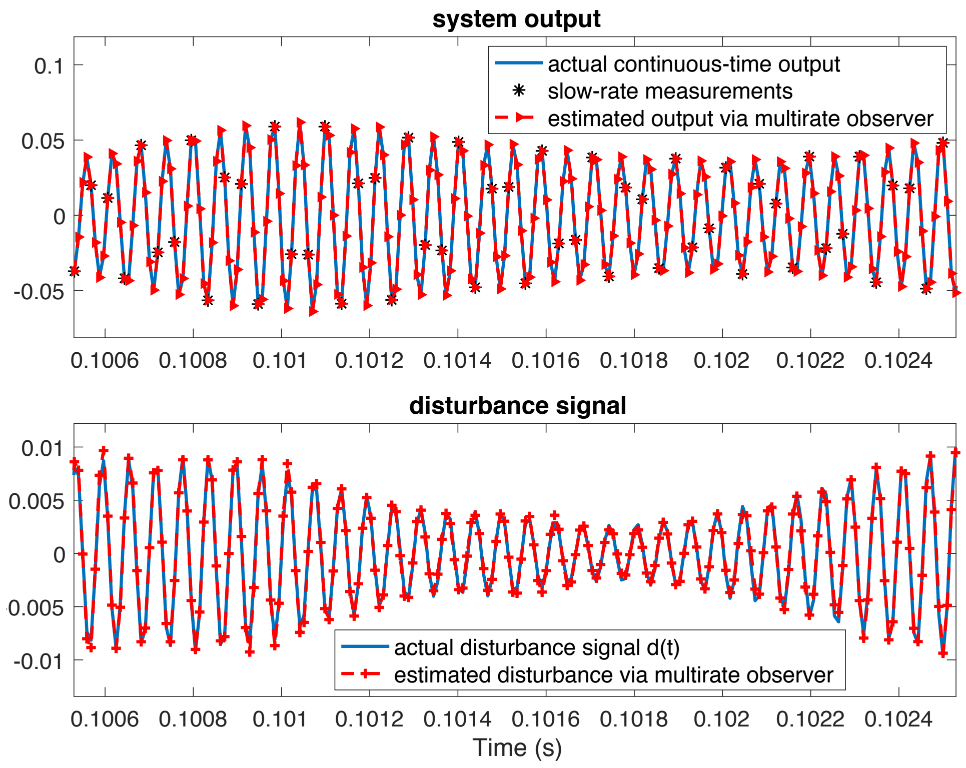


**Figure 9.6:** System output and disturbance estimates with accurate $G_D$

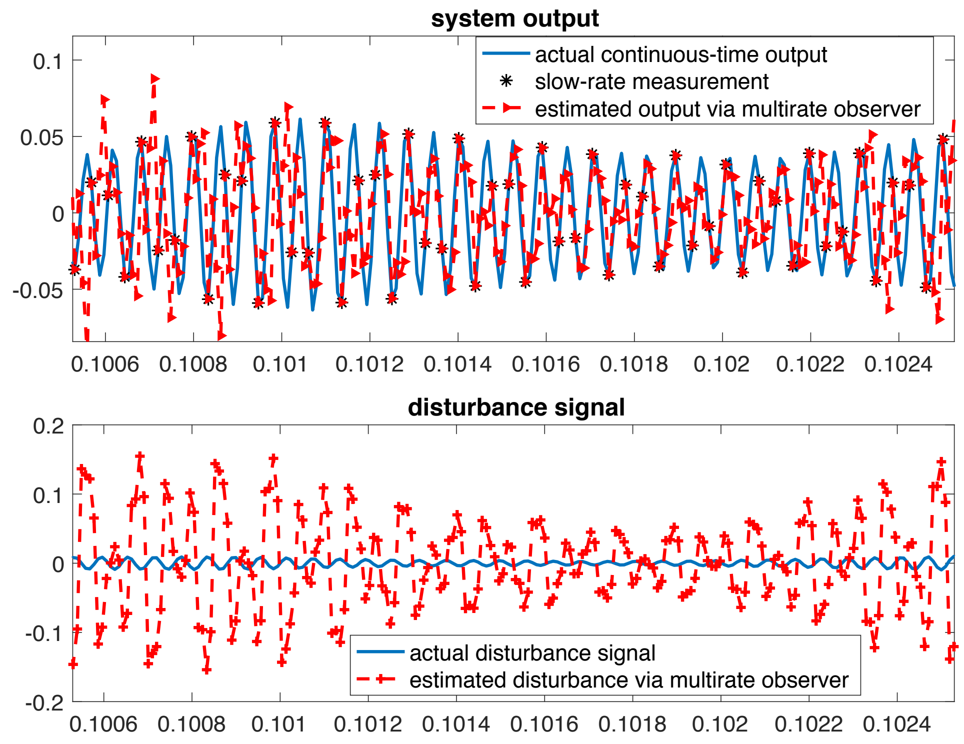


**Figure 9.7:** System output and disturbance estimates with inaccurate $G_D$

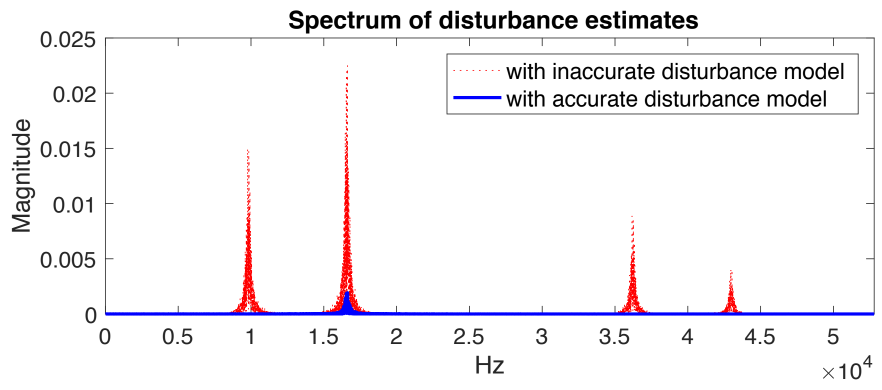


**Figure 9.8:** Spectra of $\hat{d}(kT_f)$ with accurate and inaccurate models

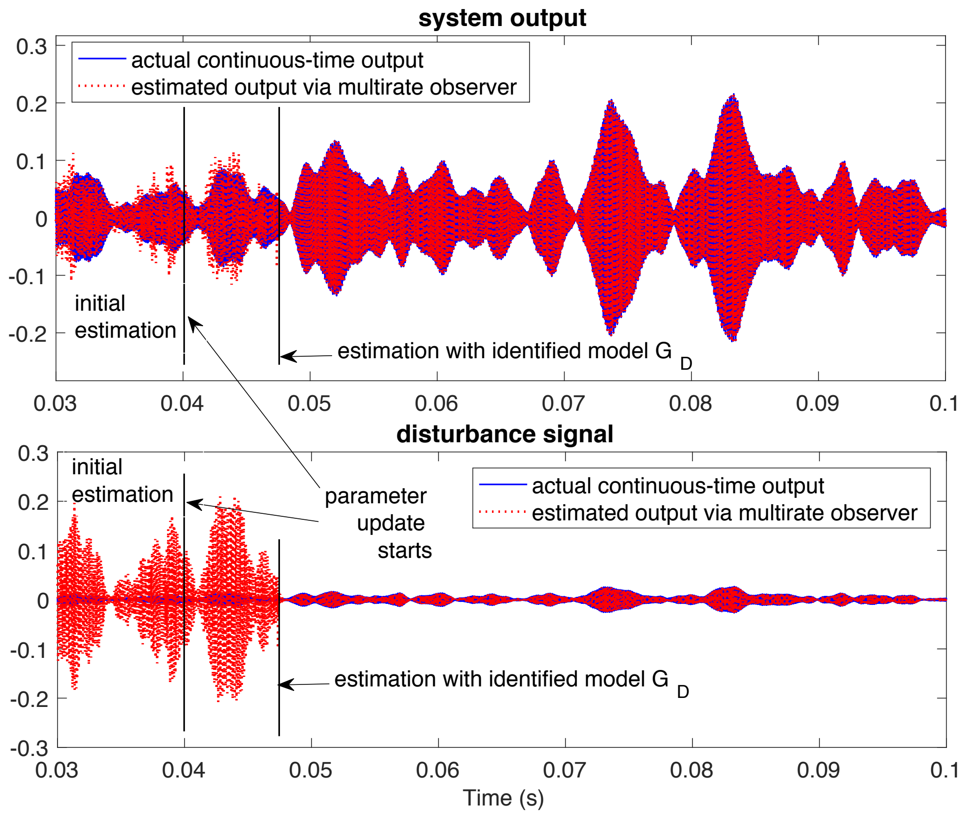


**Figure 9.9:** Estimates of output and disturbance with the proposed two-step procedure

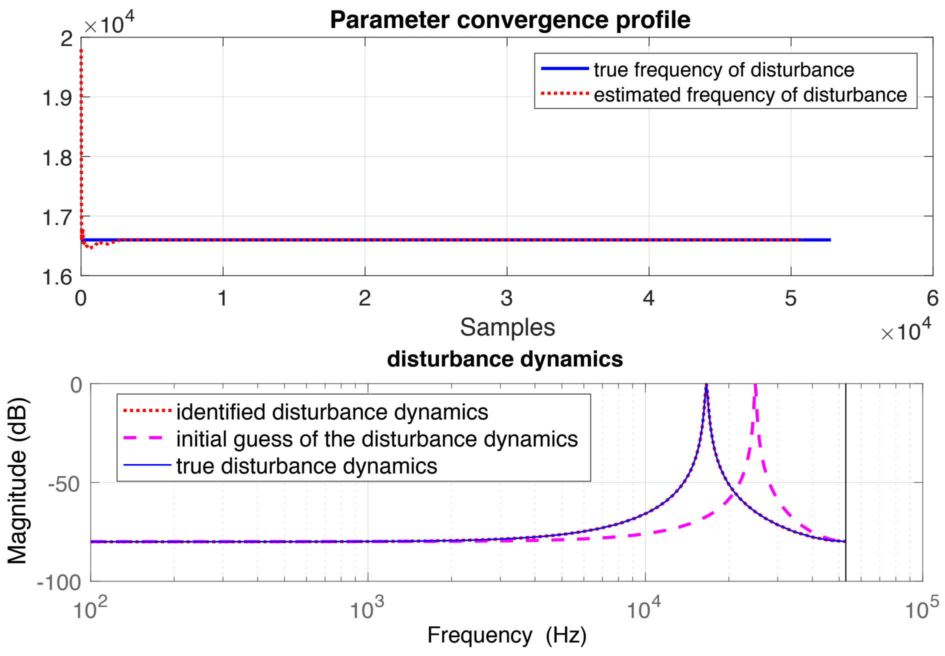


**Figure 9.10:** Converged parameter and the identified $G_D$

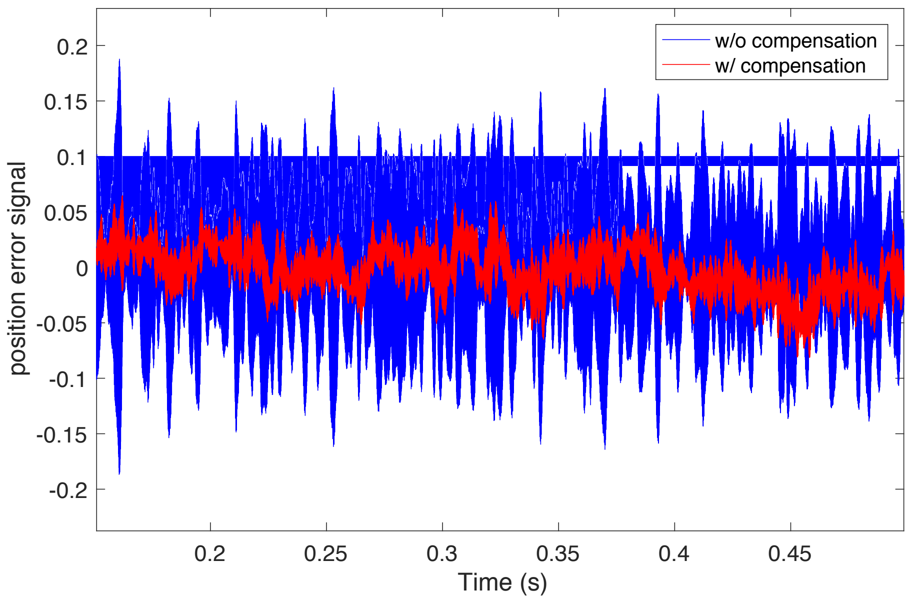


**Figure 9.11:** Position error signal

# Chapter 10

# Selective Iterative Learning Control

In Chapter 10 and Chapter 11, we will discuss the utilization of selective and adaptive learning control to enable high-performance low-level machine behavior in the presence of model uncertainties and external disturbances. Autonomous systems need to interact with a dynamic environment, and the high-level behavior might generate a variety of commands for the low-level controllers. Traditional controller designs cannot predict every possible scenario and tune the controllers' parameters in advance. Moreover, due to limited bandwidth, the achievable performance of feedback controllers is limited. We would like to explore more flexibility in the controller design, leveraging the power of learning from experience and data. Iterative learning control (ILC) is such a controller. Unlike traditional feedback control, it utilizes the repeatability of tasks and learns unknown models from training information over time. A feedforward signal is synthesized offline to improve the control performance.

It has been widely verified that ILC works well to handle repetitive uncertainties and disturbances in repetitive tasks [137, 18, 10, 22]. However, in practice, particularly for autonomous systems, the uncertainties and disturbances are not necessarily repetitive. Under such circumstances, traditional ILC might fail to achieve the required performance.

*Our key insight for these two chapters is that when both repetitive and non-repetitive uncertainties and disturbances exist, the learning loop in ILC should be re-designed to assure that performance does not degrade. We enhance it via selective learning and adaptation.*

Chapter 10 will focus on selective learning and adaptation will be discussed in Chapter 11.

## 10.1 Introduction

A great deal of efforts have been made to attain robust ILC performance in the presence of non-repetitive disturbances. One approach is to prefilter the input error signal and decompose it into a repetitive part and a non-repetitive part. Only the repetitive component is allowed to enter the learning loop of ILC. For example, Lee et. al [68] and Phan and Longman [103] used Kalman filters to remove the iteration-independent components for more effective learning, while Merry et. al [88] employed a wavelet filter to filter out the non-repetitive disturbances. Other approaches focus on adjusting the ILC scheme itself. Examples of this class include high-order ILC, segmented ILC, and ILC with a time-varying robustness filter (Q filter). More specifically, assuming that the patterns of the non-repetitiveness are known, Chen and Moore [28] constructed an iteration-domain disturbance observer (a special high-order ILC) for non-repetitive disturbance rejection. Sandipan et. al [89] used a time-domain segmented ILC strategy for the precision motion control of a wafer scanner, where the learning process is turned on and off based on the magnitudes of the repetitive and non-repetitive disturbances in each iteration. A further generalization is to equip ILC with a time-varying Q filter whose cut-off frequency is iteratively tuned online according to the time-frequency analysis of the positioning errors. When repetitive disturbances are located mainly at low frequencies, the Q-filter bandwidth is decreased to reduce the influence of the non-repetitive disturbances. Otherwise the bandwidth is increased for maximum learning ability and better performance. Zhang et. al [162] and Rotariu et. al [116] both constructed such ILC schemes; [162] used a wavelet transform for time-frequency analysis and [116] adopted the Wigner distribution algorithm. The aforementioned ILC algorithms with enhanced robustness can effectively avoid undesired error amplifications caused by the non-repetitive disturbances, yet little attention has been paid to the non-repetitive disturbance rejection performance.

*We propose new ILC scheme combining a disturbance observer (DOB) and a time-varying Q filter is proposed for performance improvement in the presence of non-repetitive disturbances, especially when their frequencies overlap with the repetitive disturbances.*

## 10.2 Experiment Setup

A wafer scanner is a machine that performs the essential photolithography steps in the manufacture of integrated circuits. It consists of a light source, a reticle stage, several projection lenses and a wafer stage, as shown in Fig. 10.1. The wafer stage and the reticle stage are both high precision motion systems that carry a silicon wafer and a mask with designed circuits patterns. The tolerable positioning errors of the two stages are in the order of nano-meters so that the patterns can be accurately printed on the wafer. A laboratory testbed wafer scanner is shown in Fig. 10.2. The stages here are both driven by three-phase linear motors and positions of the stages are measured by a laser interferometry system at a sampling frequency of 2.5kHz. We consider the reference tracking problem for the wafer stage. An example of the scanning trajectory is shown in Fig. 10.3.

Figure 10.4 shows the measured and the identified closed-loop frequency response from the reference input $y_d(k)$ to the output $y(k)$ of the wafer scanner (plant $P(z^{-1})$)

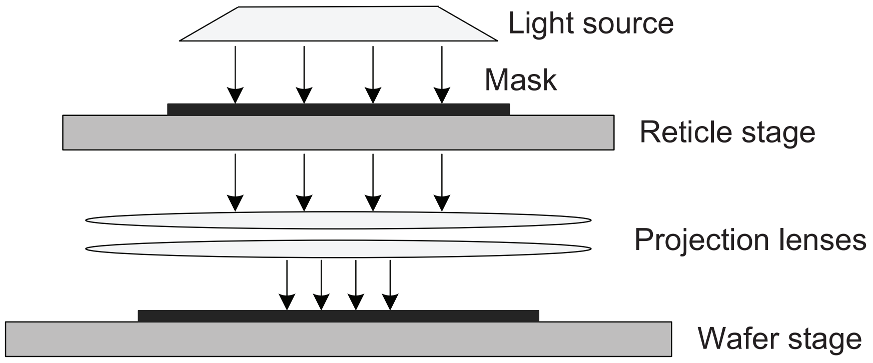


**Figure 10.1:** Schematic of the photo-lithography process

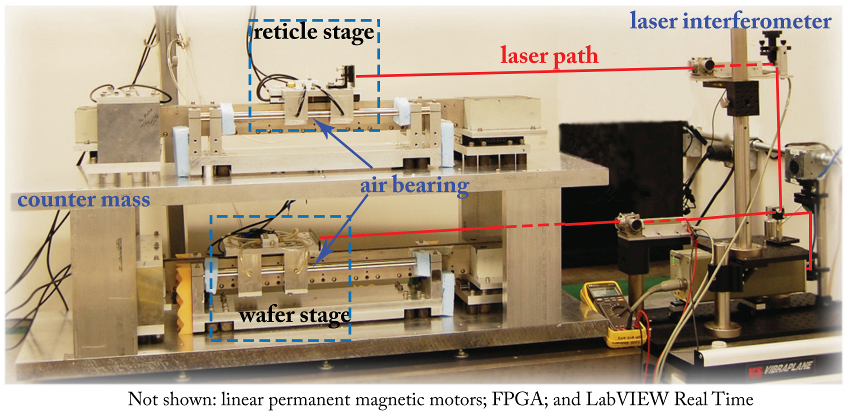


**Figure 10.2:** The experimental hardware of the testbed wafer scanner

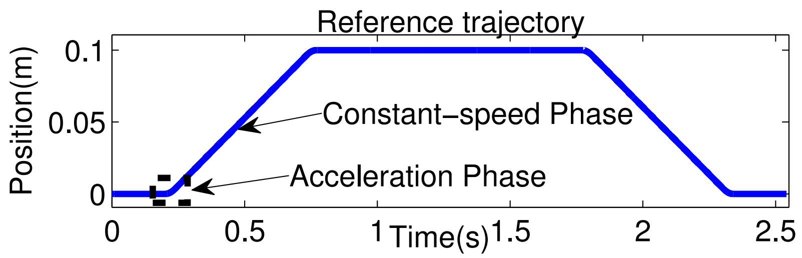


**Figure 10.3:** Reference trajectory of the wafer stage in one scan

with a baseline PID feedback controller $C(z^{-1})$ (See Fig. 10.5). The identified nominal model $P_n(z^{-1})$ of the plant $P(z^{-1})$ is:

$$P_n(z^{-1}) = 3.4766 \times 10^{-7} z^{-2} \frac{1 + 0.8z^{-1}}{1 - 2z^{-1} + z^{-2}} \tag{10.1}$$

where $z^{-1}$ denotes the one-step delay operator.

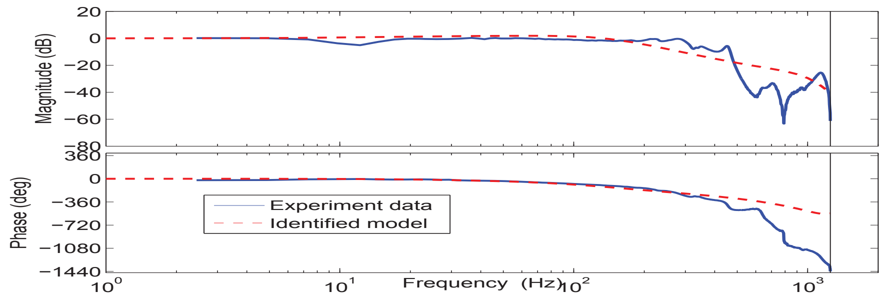


**Figure 10.4:** Measured and identified closed-loop frequency response from $y_d(k)$ to $y(k)$ of the system

## 10.3 Problem Formulation with Standard ILC

### 10.3.1 Standard ILC

Figure 10.5 shows a serial ILC structure added to the feedback control loop. $y_d(k)$, $r_j(k)$, $y_j(k)$, $e_j(k)$, $u_j(k)$ and $d_j(k)$ represent, respectively, the reference signal, the feedforward signal from ILC, the output position signal, the position error signal, the input control signal and the disturbance signal in the j-th iteration.

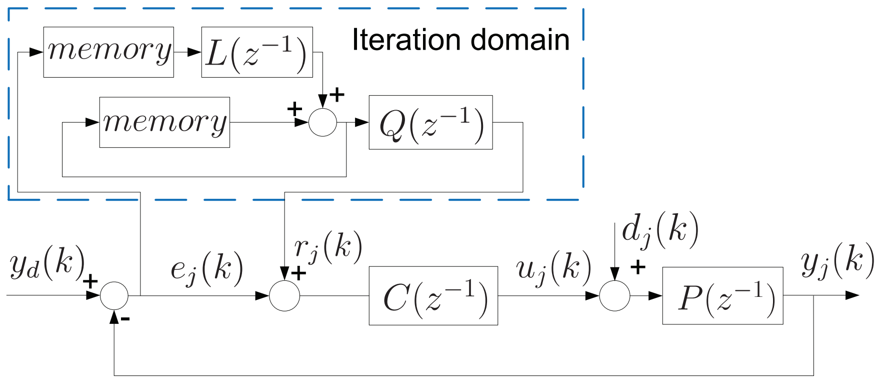


**Figure 10.5:** A serial ILC added to the feedback loop

Decompose the disturbance $d_j(k)$ in Fig. 10.5 to a repetitive disturbance $d_r(k)$ and a non-repetitive one $d_{n_j}(k)$. The system output can then be written as:

$$\begin{aligned} Y_j(z^{-1}) &= \frac{P(z^{-1})C(z^{-1})}{1+P(z^{-1})C(z^{-1})}[Y_d(z^{-1})+R_j(z^{-1})] \\ &\quad + \frac{P(z^{-1})}{1+P(z^{-1})C(z^{-1})}[D_r(z^{-1})+D_{n_j}(z^{-1})] \end{aligned} \tag{10.2}$$

where the sensitivity function and complementary sensitivity function of the feedback loop are $S_0(z^{-1}) = 1/[1+P(z^{-1})C(z^{-1})]$ and $T_0(z^{-1}) = P(z^{-1})C(z^{-1})/[1+P(z^{-1})C(z^{-1})]$; capitalized symbols $Y_j$, $Y_d$, $R_j$, $D_r$ and $D_{n_j}$ are used for expressing the signals in the $z$

domain. Then the feedforward command $r_j(k)$ generated in a standard first-order ILC is:

$$r_j(k) = Q(q)[r_{j-1}(k) + L(q)e_{j-1}(k+m)] \tag{10.3}$$

where $k$ is the time index within each iteration, $m$ is the relative degree of $T_0(z^{-1})$, and $q$ represents the forward time-shift operator, i.e., $qx(k) = x(k+1)$. $Q(q)$ and $L(q)$ are the robustness filter and the learning filter in ILC, respectively. Substituting (10.3) into (10.2) yields the closed-loop iteration-domain dynamics:

$$\begin{aligned} R_{j+1} &= Q(1-z^m LT_0)R_j + z^m QL[(1-T_0)Y_d - PS_0(D_r + D_{n_c j})] \\ E_{j+1} &= Q(1-z^m LT_0)E_j + (1-Q)[(1-T_0)Y_d + PS_0 D_r] \\ &\quad + PS_0(D_{n_{j+1}} - QD_{n_j}) \end{aligned} \tag{10.4}$$

Thus, a sufficient condition for stability of the iteration process in (10.4) is that $Q(1-z^m LT_0)$ satisfies:

$$\|Q(1-z^m LT_0)\|_\infty < 1 \tag{10.5}$$

where $\|\bullet\|_\infty = \max\limits_{0\leqslant\omega\leqslant\pi} |\bullet|_{z=e^{j\omega}}$. With (10.5) satisfied, performance of the standard ILC is evaluated by the asymptotic errors which can be expressed as:

$$\begin{aligned} E_\infty &= \frac{1-Q}{1-Q(1-z^m LT_0)}[(1-T_0)Y_d + PS_0 D_r] \\ &\quad + \frac{1}{1-Q(1-z^m LT_0)} PS_0(D_{n_{j+1}} - QD_{n_j}) \end{aligned} \tag{10.6}$$

Equation (10.5) reveals that an optimal choice for the learning filter L is the inverse of the closed-loop complementary sensitivity function, namely, $L = z^{-m}T_0^{-1}$. Here $z^{-m}$ guarantees that L is realizable. More details about this will be given in Section 10.3.2. Equation (10.6) also indicates that for perfect error rejection, namely, for eliminating the repetitive errors in one iteration, Q should be equal to one at all frequencies. However, due to the model mismatches at high frequencies, the Q filter is normally set as a lowpass filter whose bandwidth is determined by uncertainties of the system model. For example, if the actual system is:

$$P(z^{-1}) = P_n(z^{-1})(1 + \triangle(z^{-1})) \tag{10.7}$$

where $P_n(z^{-1})$ is the identified nominal model in (10.1) and $\triangle(z^{-1})$ is the multiplicative uncertainty term, then $T_0$ can be expressed as:

$$T_0(z^{-1}) = T_{0n}(z^{-1})(1 + \triangle_T(z^{-1})) \tag{10.8}$$

where $T_{0n}(z^{-1}) = P_n(z^{-1})C(z^{-1})/(1 + P_n(z^{-1})C(z^{-1}))$ and $\triangle_T(z^{-1})$ is the equivalent uncertainty in the complementary sensitivity function. The learning filter then becomes

$L = z^{-m}T_{0n}^{-1}$. Recall (10.5) and we will get the following condition for stability robustness:

$$|Q(e^{j\omega})| < \frac{1}{|\triangle_T(e^{j\omega}))|}, \forall\omega \tag{10.9}$$

Suppose the cut-off frequency of the Q filter is $\omega_c(Q)$, then only repetitive errors at frequencies lower than $\omega_c(Q)$ will be learned and attenuated by ILC, while for those at higher frequencies, learning is essentially cut off. However, in practice, repetitive errors may be distributed over a wide frequency range (for example, the case for the position errors of the wafer scanner in the acceleration phase). Also, non-repetitive errors contain components at frequencies lower than $\omega_c(Q)$(such as position errors caused by the force ripple in the wafer scanner). Therefore, standard ILC may fail to learn some repetitive errors and in the meantime mistakenly learn some non-repetitive ones, resulting in error amplifications. Consider, for instance, a pure sinusoidal non-repetitive periodic disturbance with fixed frequency $\omega_0$ that is smaller than $\omega_c(Q)$ but with a random initial phase in each iteration, i.e., $d_{n_j}(k) = sin(\omega_0 k + \phi_j)$. Then $Q(e^{j\omega_0})$ is approximately 1 and the error dynamics are:

$$e_{j+1}(k) = [1 - q^m L(q)T_0(q)]e_j(k) + P(q)S_0(q)[d_{n_{j+1}}(k) - d_{n_j}(k)]$$

where

$$d_{n_{j+1}}(k) - d_{n_j}(k) = 2\cos(\omega_0 k + \frac{\phi_j + \phi_{j+1}}{2})\sin(\frac{\phi_{j+1} - \phi_j}{2})$$

Thus, in the worst case, amplification of disturbance by a factor of two may be caused in the learning process.

### 10.3.2 Experimental Results with Standard ILC

Figure 10.6 and Fig. 10.7 show the experiment results of performing a standard ILC on the wafer scanner. The system has a non-repetitive disturbance at about 18.32Hz caused by the force ripple of the linear motor. The L and Q filters are designed according (10.5) and (10.9), $L(z^{-1}) = z^{-m}T_{0n}^{-1}(z^{-1}) = z^{-m}[1 + P_n(z^{-1})C(z^{-1})]/[P_n(z^{-1})C(z^{-1})]$. In this system, $T_{0n}(z^{-1})$ is minimum phase, so $L(z^{-1})$ can be directly derived, otherwise $L(z^{-1})$ should be designed using stable inversion method such as the ZPET algorithm [143]. The Q filter is a lowpass filter with cut-off frequency $\omega_c = 300\pi rad/s$. It can be seen in Fig. 10.6 that from the first to the second iteration, the positioning performance is significantly improved. However, no improvement is apparent in the following iterations. Figure 10.7 shows the frequency spectrum of the position errors in iterations 2 to 5. The figure shows that the non-repetitive disturbance at about 18.32Hz is greatly amplified, which has limited the performance improvement of ILC, as confirmed in Fig. 10.8.

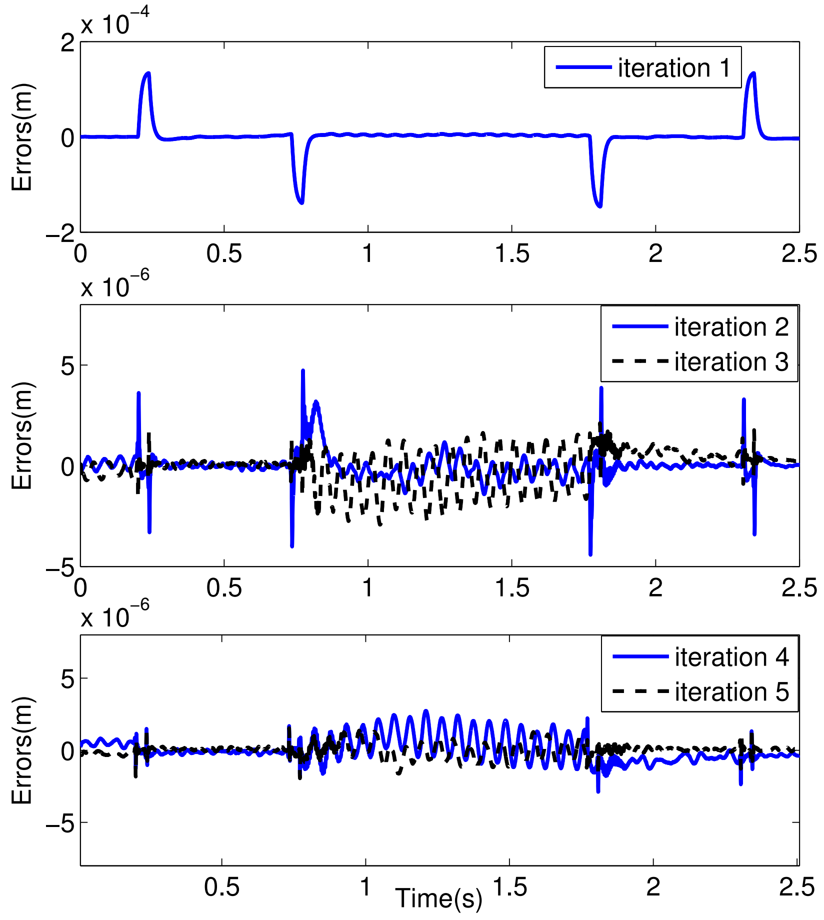


**Figure 10.6:** Positioning error signals in different iterations

## 10.4 Selective ILC with Non-Repetitive Disturbances

### 10.4.1 Structure of Selective ILC

To enhance the performance robustness of ILC in the presence of non-repetitive disturbances, a new ILC scheme is proposed in this section, as shown in Fig. 10.9. It contains two features: the selective iterative learning process (the dashed box) and a DOB for non-repetitive disturbance rejection (the dotted box). $Q_I$ and $Q_D$ represent the Q filters in ILC and in DOB, respectively. $P_m^{-1}$ is a stable and realizable inverse of the nominal model $P_n(z^{-1})$ such that $P_m^{-1}(z^{-1}) = z^{-m}P_n^{-1}(z^{-1})$.

The first benefit of this ILC scheme is enhanced attenuation of both repetitive disturbance and non-repetitive disturbance, even if their frequencies overlap with each other. On one hand, non-repetitive disturbances at frequencies within the bandwidth of the DOB is greatly rejected, generating a more "clear signal" in time domain. On the other hand, the time-varying Q filter further controls the remaining non-repetitive errors

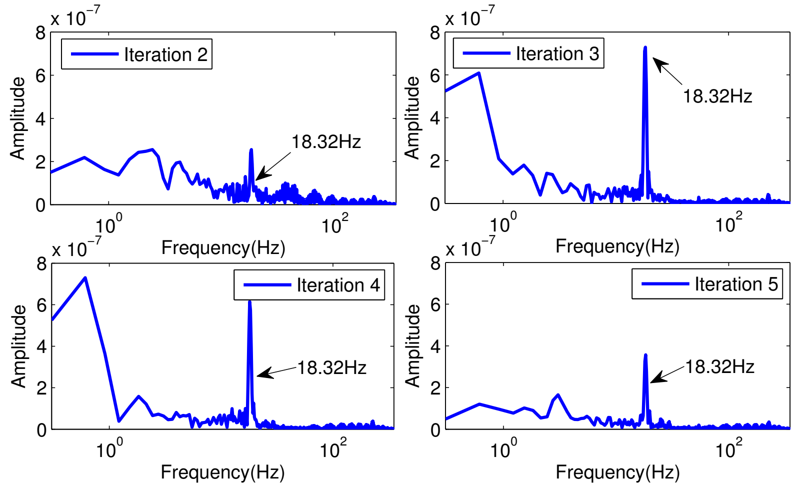


**Figure 10.7:** Frequency-domain tracking errors for iteration 2 to 4

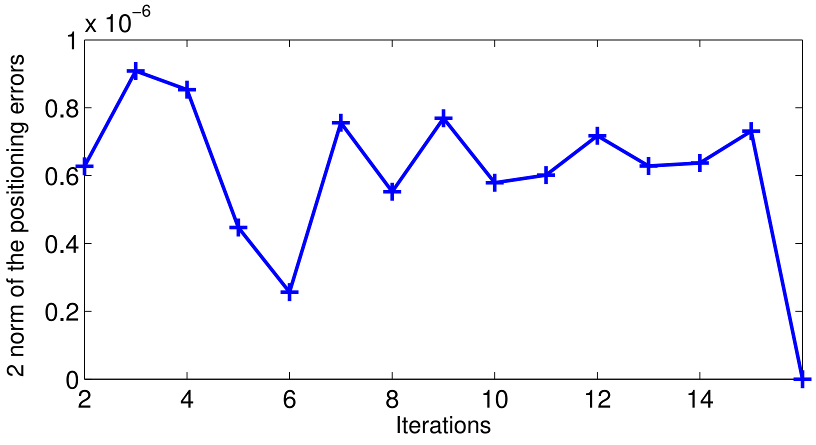


**Figure 10.8:** 2-norm of the tracking errors from iteration 2 to 16

from entering the learning loop of ILC in iteration domain and maximally increases the repetitive-disturbance-attenuation bandwidth in different phases of the trajectory. Moreover, the proposed ILC sheme provides more design flexibility to the design of $Q_I$ and $Q_D$ as discussed in the following section.

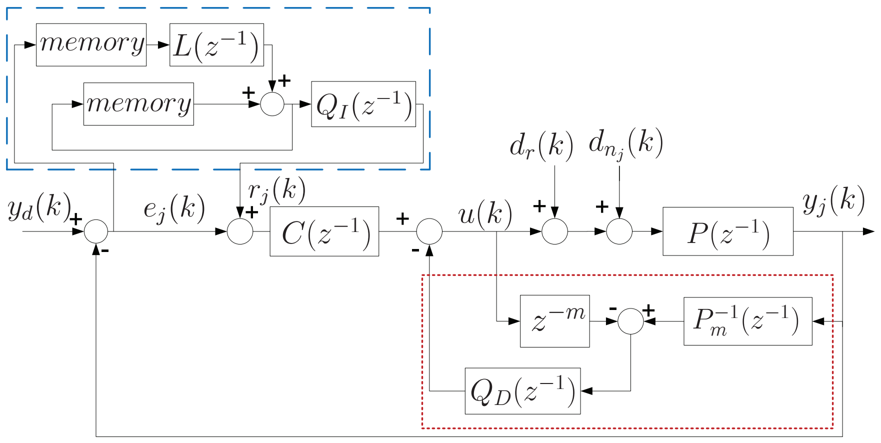


**Figure 10.9:** The proposed selective ILC with DOB

### 10.4.2 Closed-loop Dynamics and Design of $Q_I$ and $Q_D$

As shown in Fig. 10.9, the closed-loop positioning error *in the first iteration* is:

$$\begin{aligned} e_1(k) &= y_d(k) - y_1(k) \\ &= \frac{1 - Q_D(q^{-m} - PP_m^{-1})}{1 + PC - Q_D(q^{-m} - PP_m^{-1})} y_d(k) \\ &\quad - \frac{(1 - q^{-m}Q_D)P}{1 + PC - Q_D(q^{-m} - PP_m^{-1})}[d_r(k) + d_{n_1}(k)] \end{aligned} \tag{10.10}$$

where we have the new sensitivity function $S(z^{-1})$ and complementary sensitivity function $T(z^{-1})$ as:

$$S = \frac{1 - q^{-m}Q_D}{1 + PC - Q_D(q^{-m} - PP_m^{-1})} \tag{10.11}$$

$$T = \frac{PC + Q_D PP_m^{-1}}{1 + PC - Q_D(q^{-m} - PP_m^{-1})} \tag{10.12}$$

Based on (10.10) and recalling (10.2) - (10.6), we can then derive the new closed-loop iteration-domain ILC dynamics:

$$\begin{aligned} R_{j+1} &= Q_I(1 - z^m LT)R_j + Q_I z^m L[(1-T)Y_d - PS(D_r + D_{n_j})] \\ E_{j+1} &= Q_I(1 - z^m LT)E_j + (1 - Q_I)[(1-T)Y_d + PSD_r] \\ &\quad + PS(D_{n_{j+1}} - Q_I D_{n_j}) \end{aligned} \tag{10.13}$$

The goal of ILC is to cancel all repetitive disturbances by learning. Recalling (10.5), by choosing $L = z^{-m}T_n^{-1}$, we can design a lowpass filter $Q_I$ with cut-off frequency $\omega_c$ that satisfies $|Q_I(e^{j\omega})| < \dfrac{1}{|\triangle_T(e^{j\omega})|}, \forall\omega$. Noting $1 - Q_I(e^{j\omega}) \approx 0$ at low frequencies

($\omega < \omega_c$) in (10.13), the low-frequency repetitive disturbances can be almost canceled in one iteration. In the meantime, the non-repetitive component of the error in (10.13) caused by the non-repetitive disturbances is

$$e_{n_{j+1}}(k) = P\frac{1 - q^{-m}Q_D}{1 + PC - Q_D(q^{-m} - PP_m^{-1})}n_j(k) \tag{10.14}$$

where $n_j = d_{n_{j+1}}(k) - Q_I dn_j(k)$.

As discussed in Section 10.2, for the case where the non-repetitive disturbances $d_{n_j}(k)$ are concentrated at particular frequencies with iteration-dependent amplitudes and phases, undesired amplification will occur with $Q_I(e^{j\omega}) \approx 1$ at those frequencies. Thus, more consideration is required for the design of $Q_I$ and will be given in Section 10.4.2.

### Non-repetitive Disturbances at Frequencies Within the Bandwidth of $Q_D$

In this case, the non-repetitive disturbance $n_j(k)$ at frequencies within the bandwidth of $Q_D$ will be rejected by DOB if $Q_D$ is designed to satisfy the following constraints:

$$\begin{cases} |Q_D(e^{j\omega})| < \dfrac{1}{|\triangle_T(e^{j\omega})|}, \forall\omega \\ |1 - z^{-m}Q_D|_{z=e^{j\omega_{n_j}}} \ll 1 \quad \omega_{n_j} \leqslant \omega_c(Q_D) \end{cases} \tag{10.15}$$

where $\triangle_T(e^{j\omega})$ is the multiplicative uncertainty term of the complementary sensitivity function in (10.8). The first constraint guarantees the closed-loop stability with DOB **[KempfAIM1996]** and makes $Q_D$ a low-pass filter because $\triangle_T(e^{j\omega})$ is normally large at high frequencies. The second constraint comes from (10.14) and gives us an additional guideline for designing $Q_D$ properly. Here, to be able to compensate disturbances in discrete-time systems with time delays (e.g., the wafer scanner system), $Q_D$ should be designed carefully to estimate the amplitude of the disturbance as well as to compensate for the phase delay. For example, if $n_j$ is concentrated at low frequencies, then a proper choice for $Q(q^{-1})$ is set $1 - q^{-m}Q_D(q) = H_D(q)$ to be of high-pass characteristics. Assume that

$$1 - q^{-m}Q_D(q) = H_D(q) = \frac{B(q)}{A(q)}J(q) \tag{10.16}$$

which is equivalent to the Diophantine equation, $A(q) = B(q)J(q) + q^{-m}B_{Q_D}(q)$, if we let $Q_D(q)$ share the same denominator with $H_D(q)$, namely, $Q_D(q) = B_{Q_D}(q)/A(q)$. Solving this Diophantine Equation gives us a minimum-order $Q_D(q)$ satisfying (10.16).

As an example, let

$$H_D(q) = \frac{0.9481 - 1.896q^{-1} + 0.9481q^{-2}}{1 - 1.894q^{-1} + 0.899q^{-2}}$$

This filter has a cut-off frequency at 30Hz and the resultant Diophantine Equation yields

$$Q_D(q) \quad = \quad \frac{0.1118 - 0.1064q^{-1}}{1 - 1.894q^{-1} + 0.899q^{-2}}$$

with $J(q) = 1.055 + 0.1122q^{-1}$. The frequency response of $Q_D(z^{-1})$ and $H_D(z^{-1})$ are shown in Fig. 10.10.

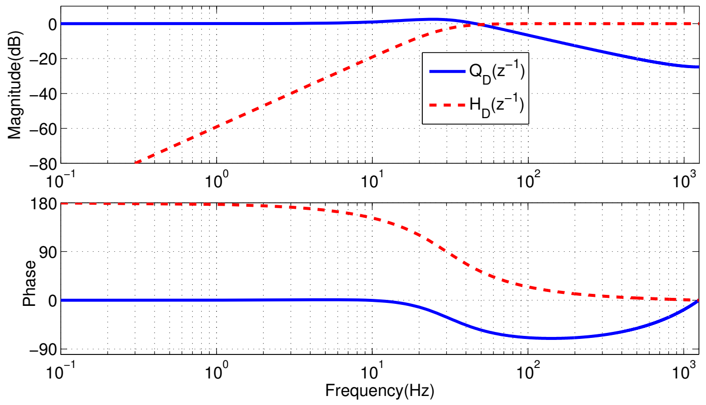


**Figure 10.10:** The designed $Q_D(z^{-1})$ and the resultant $H_D(z^{-1})$

## Non-repetitive Disturbances at Frequencies Above the Bandwidth of $Q_D$

Non-repetitive disturbances at frequencies above the bandwidth of $Q_D$ cannot be rejected by DOB, and a proper $Q_I$ is required which can selectively prevent the non-repetitive disturbances from entering the ILC learning scheme and maximally preserve its ability to reject repetitive disturbances.

Suppose that there is a sinusoidal non-repetitive periodic disturbance component $d_n(k)$ at frequency $\omega_0$ which is much higher than the bandwidth of $Q_D$, i.e., $Q_D(\omega_0) \ll 1$ in (10.14). Then the tracking error in the $(j+1)$-th iteration will become

$$e_{n_{j+1}}(k) = PS[d_{n_{j+1}}(k) - Q_I d_{n_j}(k)] \tag{10.17}$$

Therefore, to prevent the disturbance in the j-th iteration from entering the $(j+1)$-th iteration, $Q_I(e^{j\omega_0}) \approx 0$ should be satisfied. Namely, $Q_I$ should contain a notch filter as follows:

$$Q_I(q) = Q_{I_0}(q)\frac{1 - 2\cos(2\pi\omega_0)q^{-1} + q^{-2}}{1 - 2\alpha\cos(2\pi\omega_0)q^{-1} + \alpha^2 q^{-2}} \tag{10.18}$$

where $Q_{I_0}(q)$ is a baseline Q filter in the standard ILC.

As we discussed in Section **??**, the position error of a precision system has different characteristics in different phases of the trajectory. In the acceleration phase, the error mainly comes from tracking of the trajectory and contain rich frequency components. Thus, the higher the bandwidth of $Q_I$ ($\omega_c$), the better the learning performance. However, in the constant-speed phase, the error components caused by the non-repetitive disturbances (for example, force ripple) dominate and the ILC should selectively filter them out to avoid undesired amplification; namely, $Q_I$ in this phase should contain notches at those frequencies where the non-repetitive disturbances appear. Consequently, $Q_I$ becomes a time-varying filter which has a wider bandwidth in the acceleration phase (for better trajectory following) and multiple notches in the constant-speed phase. To guarantee the performance during the switching process, a smooth switching algorithm may be used; for example,

$$Q_I = \begin{cases} Q_{I0}, & t \leqslant t_{ae} \\ (1-\alpha(t))Q_{I0} + \alpha(t)Q_{I1}, & t_{ae} < t < t_{ae} + \triangle t \\ Q_{I1}, & t \geqslant t_{ae} + \triangle t \end{cases} \tag{10.19}$$

where $Q_{I0}$ is the $Q_I$ filter in the acceleration phase and $Q_{I1}$ in the constant-speed phase designed based on (10.18). $t_{ae}$ is the time instant when the acceleration phase ends and $\triangle t$ is the switching period. $\alpha(t)$ gradually varies from 0 to 1 as $\alpha(t) = (t - t_{ae})/\triangle t$.

## 10.5 Simulation Results

Simulations have been performed to verify the proposed algorithm. The reference trajectory is as shown in Fig. 10.3. Both repetitive and non-repetitive disturbances are introduced. Repetitive disturbances are distributed over a wide frequency range [0, 100Hz] and non-repetitive periodic disturbances appear only at specific frequencies (18.32Hz and 50Hz) with initial phases varying from iteration to iteration.

Figure 10.11 shows the error signal and its frequency spectrum with the baseline PID controller.

### 10.5.1 Using Standard ILC Algorithm

The simulation result using a standard ILC algorithm is shown in Fig. 10.12. The Q filter and learning filter are configured as described in Section 10.3.2. Figure 10.12(a) shows that ILC can effectively reduce the position error. After the second iteration, however, no consistent improvement can be seen. For example, from the second iteration to the third iteration, the position error norm is significantly amplified by ILC. Figure 10.12(b) shows the frequency spectrum of the error signals in the first three iterations of ILC. It can be seen that the repetitive disturbances are effectively eliminated, but the error becomes significant at the non-repetitive disturbance frequencies (at about 18.32Hz and 50Hz).

The same phenomena are observed in experiments. As shown in Fig. 10.6 to Fig. 10.8 in Section 10.3.2, the non-repetitive disturbance at 18.32Hz is also amplified and the ILC

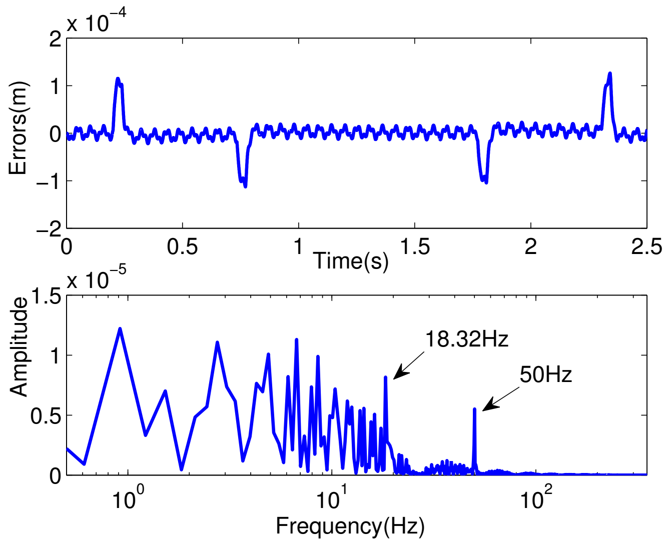


**Figure 10.11:** The position error with the baseline PID controller

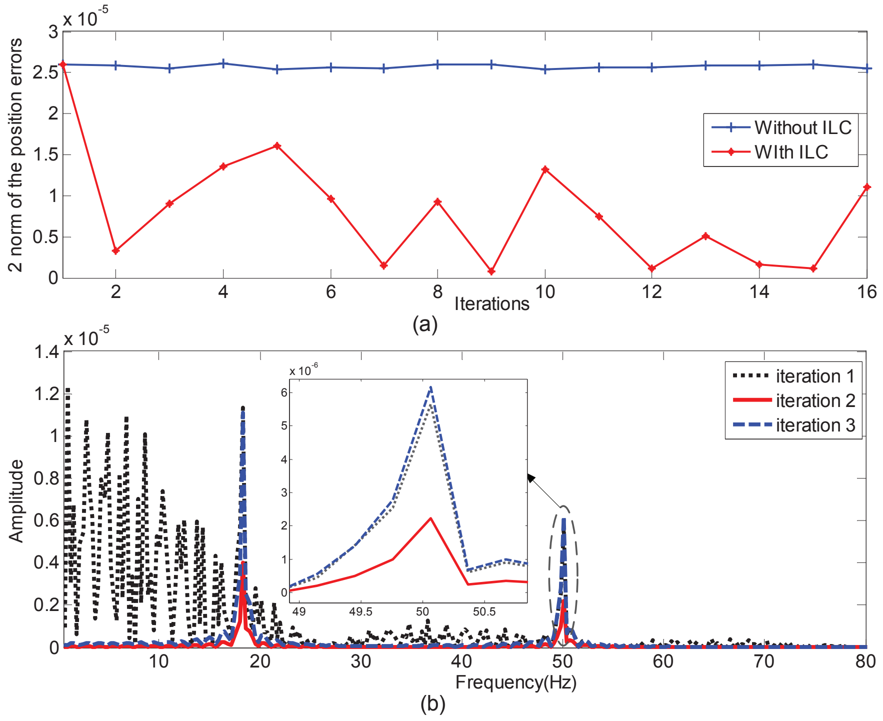


**Figure 10.12:** The position error using standard ILC without DOB in simulation

performance is degraded. Additionally, in experiments, we are subjected to more constraints in designing $Q_I$ without DOB. The robust stability condition in (10.5) requires $Q_I \approx 0$ at high frequencies, which will further degrade the ILC performance. As shown in Fig. 10.7, disturbances with frequencies higher than 150Hz ($\omega_c(Q_I)$) cannot be attenuated by learning.

### 10.5.2 Using only DOB without ILC

Figure 10.13 shows the simulation results with and without DOB in one iteration; no ILC is used. We observe that in the constant-speed phase, DOB works well and significantly reduces the position errors by suppressing the disturbances at frequencies lower than the cut-off frequency of $Q_D$ ($\omega_c(Q_D) = 60\pi \mathrm{rad/s}$). The transient performance in the acceleration and deceleration phases, however, is minimally improved. Note that in these phases, the position error caused by the tracking of the reference trajectory dominates and DOB has little effect to reduce the error. Additionally, the non-repetitive disturbance at 50Hz (larger than $\omega_c(Q_D)$) remains unchanged.

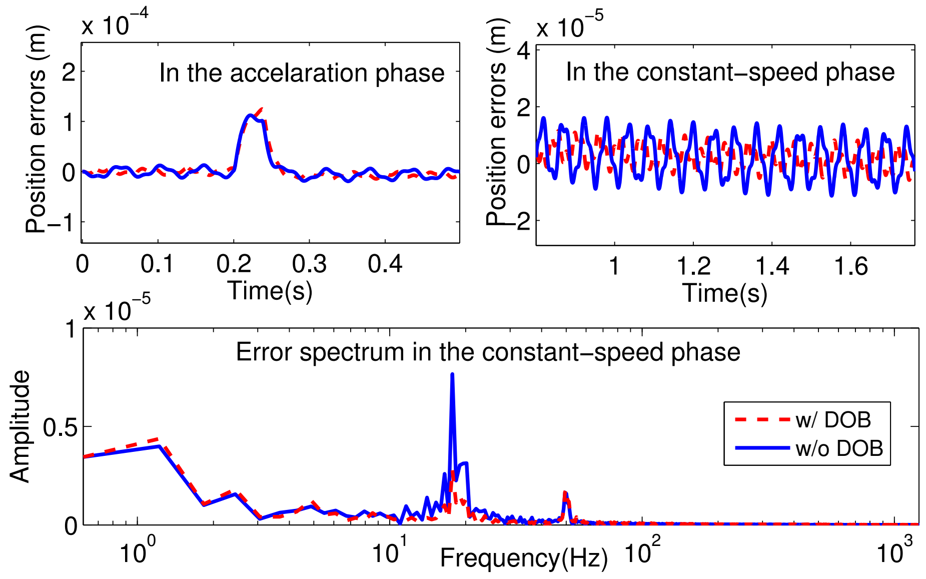


**Figure 10.13:** Position errors with and without DOB in simulation

### 10.5.3 Using the proposed ILC structure

Figure 10.14 and Fig. 10.15 show a comparison of the simulation results using different algorithms. The $Q_D$ and $Q_I$ in the proposed ILC scheme are designed, respectively, as a lowpass filter with cut-off frequency $\omega_c = 60\pi \mathrm{rad/s}$ and as a time-varying filter in

(10.19), where $\triangle t = 100T_s = 0.04s$, and $Q_D(z^{-1})$, $Q_{I0}(z^{-1})$ and $Q_{I1}(z^{-1})$ are given by

$$\begin{aligned}
Q_D(z^{-1}) &= \frac{0.1118 - 0.1064z^{-1}}{1 - 1.894z^{-1} + 0.899z^{-2}} \\
Q_{I0}(z^{-1}) &= \frac{0.02792 + 0.05583z^{-1} + 0.02792z^{-2}}{1 - 1.475z^{-1} + 0.5866z^{-2}} \\
Q_{I1}(z^{-1}) &= Q_{I0}(z^{-1})\frac{1.0372(1 - 2\cos(2\pi T_s * 18.32)z^{-1} + z^{-2})}{1 - 1.98\cos(2\pi T_s * 18.32)z^{-1} + 0.9801 * z^{-2}} \\
&\quad * \frac{0.9963(1 - 2\cos(2\pi T_s * 50)z^{-1} + z^{-2})}{1 - 1.98\cos(2\pi T_s * 50)z^{-1} + 0.9801z^{-2}}
\end{aligned}$$

Also, in the simulation, $Q_I(z^{-1})$ in the standard ILC is set to be $Q_{I0}(z^{-1})$ in each iteration.

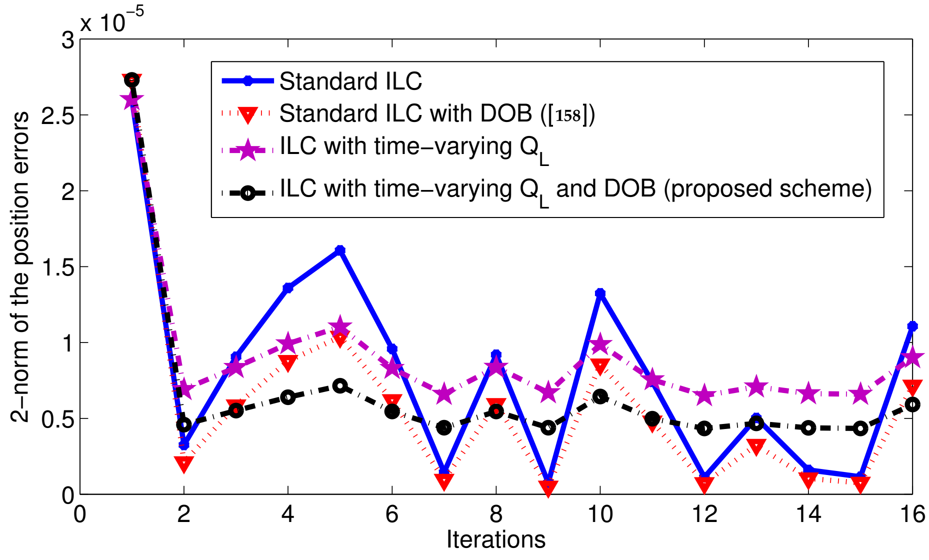


**Figure 10.14:** The position errors using different ILC algorithms in simulation

It can be seen that the proposed ILC attains the best performance with effective disturbance rejection and enhanced robustness to non-repetitive periodic disturbances. Consistent with the experimental result shown in Fig. 10.8, the position error obtained by standard ILC varies greatly in different iterations due to unexpected amplifications of the non-repetitive errors. The algorithm in [156] can significantly reduce the position error. The performance robustness to non-repetitive periodic disturbances, however, is still not good. This is because the DOB can only reduce the non-repetitive error components with frequencies lower than $\omega_c(Q_D)$ (e.g.,the error component at 18.32Hz in the simulation), but has little effect on those higher than $\omega_c(Q_D)$ (the one at 50Hz in the simulation) and undesired amplifications of those signals cannot be avoided. By introducing a time-varying $Q_L$, the proposed ILC scheme not only effectively suppresses disturbances with

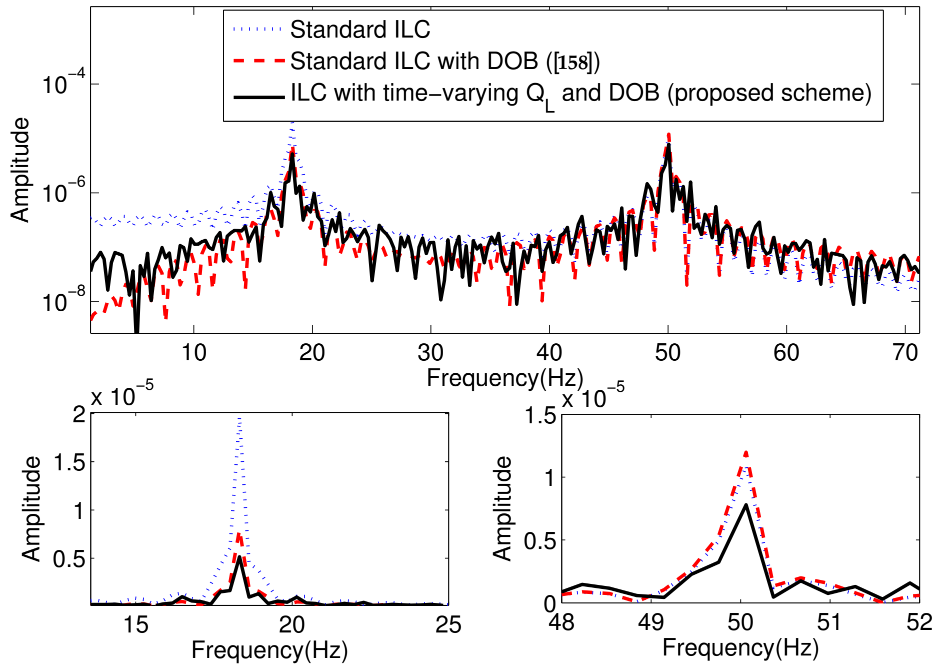


**Figure 10.15:** Spectra of the position errors in the 5-th iteration using different ILC algorithms in simulation

frequencies lower than $\omega_c(Q_D)$, but also flexibly controls the remaining non-repetitive errors from entering the learning loop of ILC. Thus, undesired amplifications of the non-repetitive disturbances have been effectively avoided. Compared to standard ILC and the algorithm in [156], it can be seen clearly in Fig. 10.15 that no amplification of the non-repetitive disturbance at 50Hz occurs by using the proposed ILC algorithm.

## 10.6 Experiment Results

We also have conducted experiments on the testbed wafer scanner system introduced in Section 2. Similar conditions are utilized as in the simulation: the reference is as shown in Fig. 10.3, repetitive disturbances over a wide frequency range ([0-100Hz]) and non-repetitive disturbances at 18.32Hz and 50Hz are introduced. Note that in real hardware, the non-repetitive disturbance at 18.32Hz is caused by the force ripple of the linear magnetic motor. The disturbance with a frequency of 50Hz is an external signal that we introduced to the system. The design of $Q_D$ and $Q_I$ are the same as in simulation.

Figure 10.16 shows the spectra of the position errors in the first and second iterations using different algorithms: the standard ILC, ILC with DOB, and selective ILC with DOB. We can see that most of the repetitive disturbances have been effectively attenuated via ILC in all three approaches. Comparing the performance between ILC and ILC with DOB, we can see that the introduction of DOB can help effectively attenuate the disturbances whose frequencies are within the bandwidth of $Q_D$, no matter being

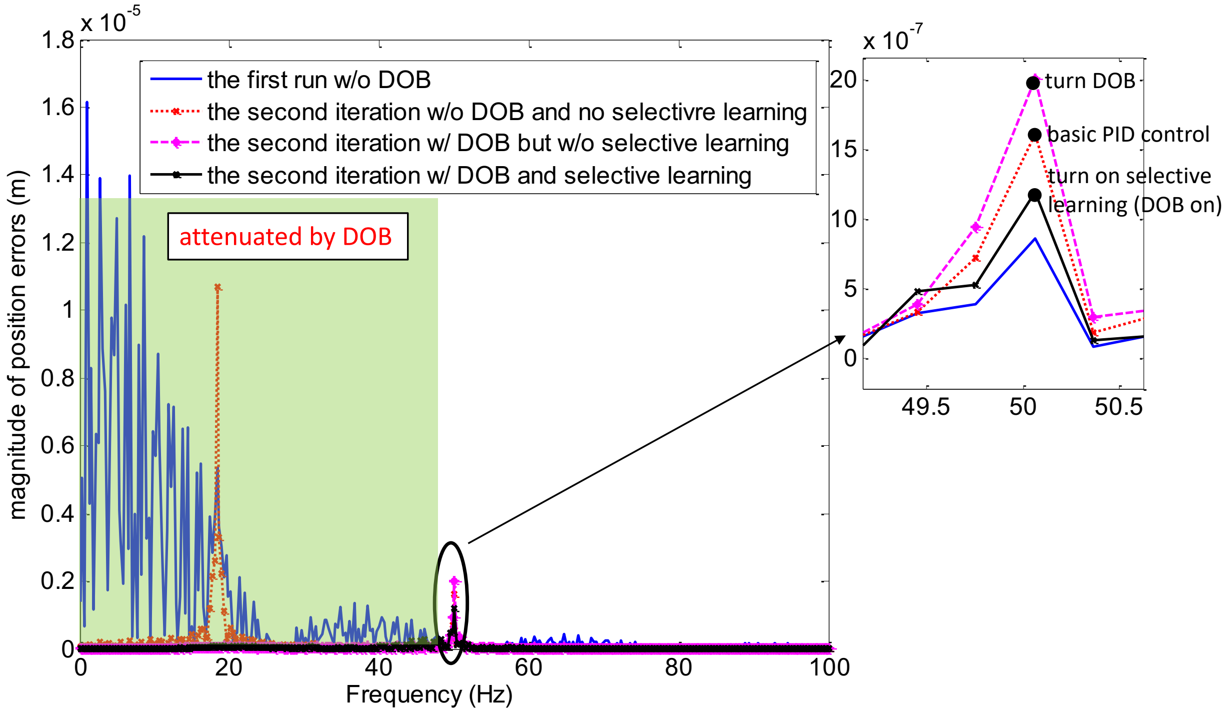


**Figure 10.16:** The spectra of the position errors in the first and second trials in experiments

repetitive or not. For the non-repetitive disturbances beyond the bandwidth of $Q_D$, the performance of ILC with DOB is degraded due to the amplification effect, as shown in the zoomed-in view on the right of Fig. 10.16. The proposed selective ILC with DOB, on the contrary, can effectively reduce such amplification and preserve the achievable performance.

Figure 10.17 gives the 2-norms of the position errors along iterations using two different ILC algorithms: ILC with DOB (blue curve) and selective ILC with DOB (red curve). We can see that starting from the same initial performance (first trial), the proposed selective ILC with DOB can achieve on average smaller position errors, reducing the 2-norms from $8.787 \times 10^{-5}\mathrm{m}^2$ to $3.0135 \times 10^{-5}\mathrm{m}^2$ . Compared to the one without selective learning, the performance converges faster and more robustly, i.e., the 2-norms of the position errors do not fluctuate from iteration to iteration.

## 10.7 Chapter Summary

In this chapter, a new ILC scheme with robust performance in the presence of non-repetitive disturbances was proposed. By integrating a DOB and a time-varying Q filter in ILC, the proposed algorithm can not only maximally preserve its repetitive-error-rejection ability, but also provide enhanced attenuation to non-repetitive disturbances at

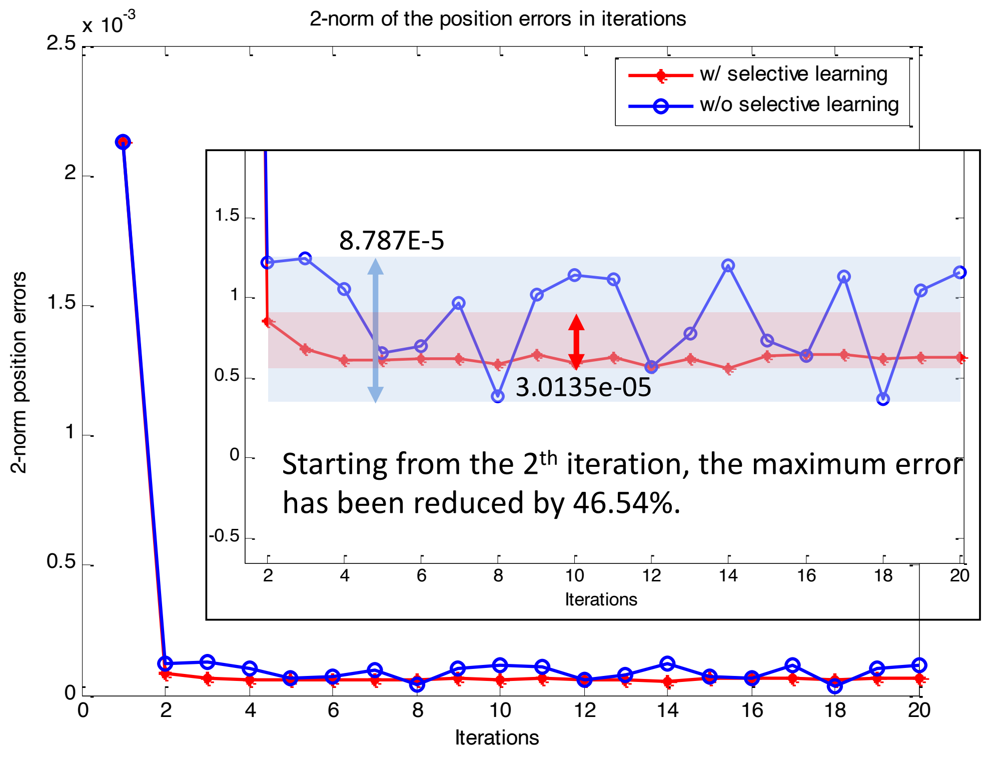


**Figure 10.17:** The 2-norms of the position errors along iterations in experiments

low frequencies. The selective learning characteristic of the time-varying Q filter avoids undesired amplification of the error caused by non-repetitive periodic disturbances. Simulation and experiment results on a wafer scanner testbed system were provided. The results showed that compared to traditional ILC approaches, the proposed selective ILC can effectiveness improve the robustness of the learning performance in the presence of non-repetitive disturbances.

# Chapter 11

# Adaptive Iterative Learning Control

As discussed in Chapter 10, disturbances and dynamic uncertainties are hardly repetitive in practice. In most systems, both iteration-varying (i.e., non-repetitive) and iteration-invariant (i.e., repetitive) disturbances exist. Iteration-varying disturbances can be categorized into two types: Type I - external state-independent disturbances (most external vibrations belong to this type), and Type II - internal state-dependent disturbances. Joint friction force in robot manipulators is a typical example of such state-dependent disturbances. The selective ILC algorithm presented in Chapter 10 focuses on addressing the Type I disturbances, and in this chapter, we will present an adaptive iterative learning control (ILC) strategy to handle the second type. In particular, we use a planar robot manipulator as an application example, and focus on the compensation of unknown but iteration-varying friction forces in joints.

## 11.1 Introduction

To deal with the non-repetitive disturbances of Type II, an adaptive ILC (AILC) approach was first proposed by [63]. Based on Lyapunov theory, AILC naturally fits in situations where the unknown disturbances are not only iteration-varying, but also state-dependent. The key idea of AILC, as discussed in [102] and [139], is to introduce an adaptive signal which iteratively identifies and compensates for the unknown disturbances and uncertain system parameters. Commonly, it requires the parameters to be constant within one iteration such as in [29], [45] and [95].

As discussed above, joint friction forces in robot manipulators, are state-dependent and time-varying disturbances. Hence, to achieve good tracking performance, we propose a novel AILC to compensate for the joint friction forces. Compared to previous formulations (e.g., [29]), we have proposed a new adaptation law based on the work in [94], which can significantly improve, not only the accuracy of parameter estimates, but also the tracking performance in joint space. Theoretical analysis and proof of error convergence are provided, along with simulation and experimental verification. Performance comparisons between a traditional AILC design in [29] and our proposed AILC are also provided, demonstrating that our proposed approach can achieve better tracking performance with improved parameter estimates when nonlinear joint friction forces exist in robot manipulators.

## 11.2 Problem Statement

### 11.2.1 Dynamic Model of Manipulators

In general, the dynamic model of an $n$-DOF robot manipulator can be expressed as

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{d} = \mathbf{u} \tag{11.1}$$

where $\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}} \in \mathbb{R}^n$ denote the vectors of positions, velocities, and accelerations of all joints, respectively. $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{n\times n}$ is the mass inertia matrix, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^{n\times n}$ is the centripetal Coriolis force matrix, and $\mathbf{G}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^n$ is the gravitational and friction force vector on the system. Uncertainty in the dynamic model may result in uncertainty of any of these three terms. $\mathbf{d} \in \mathbb{R}^n$ is the repetitive disturbance exerted on the system and $\mathbf{u} \in \mathbb{R}^n$ is the input torque. In (11.1) and throughout this chapter, all vectors and matrices are time dependent, but the time variable $t$ is omitted for brevity.

In planar manipulators, such as the one used in this work, there is no gravitational force and the term $\mathbf{G}(\mathbf{q}, \dot{\mathbf{q}})$ in (11.1) can be reduced to $\mathbf{G}(\dot{\mathbf{q}}) = \mathbf{F_{fr}}(\dot{\mathbf{q}}) \in \mathbb{R}^n$, representing the friction force vector on all $n$ joints. While many different friction models are applicable, we use the Stribeck model since it can best characterize the stiction phenomenon observed in our experiments with the robot manipulators. According to the model in [11], the Stribeck friction force on each joint can be individually expressed as

$$F_{fr_i}(\dot{q}_i) = \sqrt{2e}(F_{brk_i} - F_{c_i})e^{-(\frac{\dot{q}_i}{\nu_{st_i}})^2}\frac{\dot{q}_i}{\nu_{st_i}} + F_{c_i}\tanh(\frac{\dot{q}_i}{\nu_{coul_i}}) + f_i\dot{q}_i, \tag{11.2}$$

where $i = 1, 2, \cdots, n$. $F_{brk_i}$ and $F_{c_i}$ represent, respectively, the breakaway friction force and the Coulomb friction force. $f_i$ is the viscous friction coefficient. $\nu_{st_i}$ and $\nu_{coul_i}$ are thresholds for the Stribeck velocity and the Coulomb velocity, respectively.

### 11.2.2 Friction Compensation via Standard ILC

#### Update Law in Standard ILC

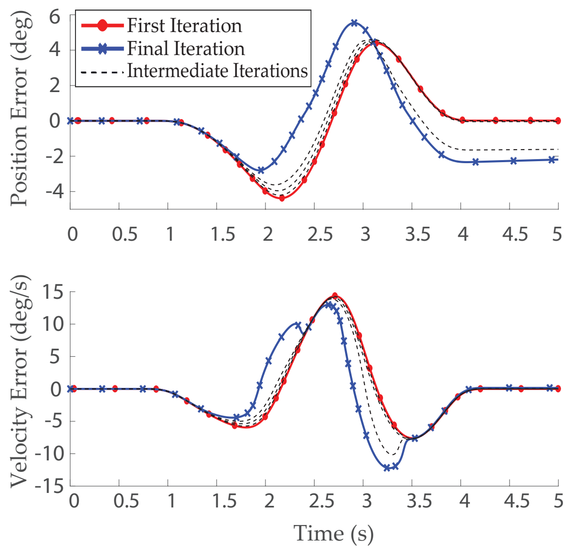
In traditional ILC algorithms, the control input is iteratively updated as:

$$\mathbf{u}^{j+1} = \mathbf{Q}^*[\mathbf{u}^j + \mathbf{L}^*\mathbf{e}^j], \tag{11.3}$$

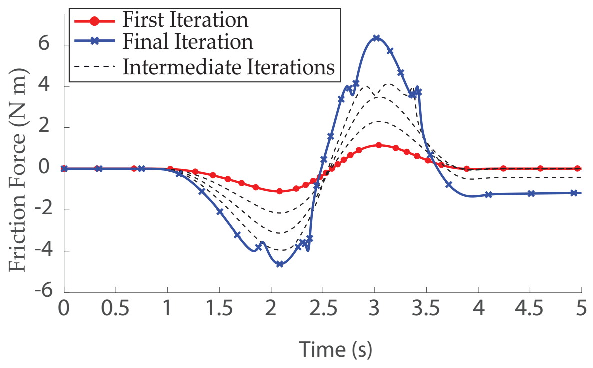
where superscript $j$ denotes the iteration index, and $\mathbf{e}^j$ is the tracking error at $j$-th iteration defined as $\mathbf{e}^j \triangleq \mathbf{q}_d - \mathbf{q}^j$ with $\mathbf{q}_d$ and $\mathbf{q}^j$ representing, respectively, the desired joint trajectory and the actual joint trajectory at j-th iteration. $\mathbf{L}^*$ represents the learning filter/matrix that varies depending on the specific control algorithm and $\mathbf{Q}^*$ represents a generic Q-filter/matrix that aims to improve the robustness of the learning process. For instance, in PD-type ILC, $\mathbf{L}^*$ is a gain matrix, and $\mathbf{Q}^*$ is typically a low-pass filter to prevent high-frequency measurement noises from deteriorating learning performance.

## Insufficient Friction Compensation via Standard ILC

To demonstrate the limitations of Standard ILC, we applied a PD-type ILC on a simulated planar 1-DOF manipulator subject to Stribeck friction. The tracking performance in joint space and the friction forces over 10 iterations are shown in Fig. 11.1. We can see that standard ILC fails to effectively compensate for significant changes in friction between iterations. Particularly at later iterations, the non-smoothness of the frictional force become more significant, resulting in a degradation in ILC performance.



**(a)** Evolution of errors



**(b)** Frictional forces

**Figure 11.1:** Joint error and friction forces over ten iterations

Considering the time-varying nature of friction and its nonlinear dependence on system states, it is difficult to compensate for this iteration-varying disturbance through standard ILC. Hence, we need to introduce an additional control signal which can identify the parameters that can efficiently represent the friction model, and incorporate those parameter estimates into the design of ILC. In the following section, a novel adaptive ILC algorithm is proposed to acheive this.

## 11.3 The Proposed Adaptive Iterative Learning Control

### 11.3.1 Control Law Design

The control input $\mathbf{u}^j$ in our proposed adaptive ILC consists of three components as follows:

$$\mathbf{u}^j = \mathbf{u}_f^j + \mathbf{u}_c^j + \mathbf{u}_l^j, \tag{11.4}$$

where $\mathbf{u}_f^j$ is the feedback input, $\mathbf{u}_c^j$ is the adaptive control input, and $\mathbf{u}_l^j$ is the iterative learning input.

The feedback control input is of PD-type, defined as

$$\mathbf{u}_f^j = \mathbf{L}(\dot{\mathbf{e}}^j + \alpha \mathbf{e}^j) \triangleq \mathbf{L}\mathbf{z}^j. \tag{11.5}$$

The matrix $\mathbf{L}$ and scalar $\alpha$ are pre-defined constants to stabilize the system.

The iterative control input is given by

$$\mathbf{u}_l^{j+1} = \mathbf{u}_l^j + \beta \mathbf{L}\mathbf{z}^j, \tag{11.6}$$

where $0 < \beta \leqslant 1$ is a constant, scalar learning gain selected based on the desired convergence rate.

The above formulation of feedback input (11.5) and iterative input (11.6) share the same formulation as in [102] and [29]. As for the adaptive control input $u_c^j$, we propose a new control law to compensate for the time-varying friction forces as follows:

$$\mathbf{u}_c^j = \hat{\mathbf{M}}(\mathbf{q}^j)\ddot{\mathbf{q}}_d + \hat{\mathbf{C}}(\mathbf{q}^j, \dot{\mathbf{q}}^j)\dot{\mathbf{q}}_d + \hat{\mathbf{G}}(\dot{\mathbf{q}}^j)\alpha(\hat{\mathbf{M}}(\mathbf{q}^j)\dot{\mathbf{e}}^j + \hat{\mathbf{C}}(\mathbf{q}^j, \dot{\mathbf{q}}^j)\mathbf{e}^j), \tag{11.7}$$

where $\hat{(\cdot)}$ denotes the estimate of a variable.

Again, for planar manipulators, we have $\hat{\mathbf{G}}(\dot{\mathbf{q}}^j) = \hat{\mathbf{F}}_{\mathbf{fr}}(\dot{\mathbf{q}}^j)$. Note that in this work, we focus on the scenarios where all the parameter uncertainties are associated with the friction forces, i.e., the mass, inertia, and Coriolis matrices are assumed to be exactly known. This allows us to rewrite (11.7) into the linear form as in (11.8) for the purposes of the adaptation law:

$$\mathbf{u}_c^j = \mathbf{Y}^j \hat{\boldsymbol{\theta}}^j, \tag{11.8}$$

where $\mathbf{Y}^j$ and $\hat{\boldsymbol{\theta}}^j$ represent the regression matrix and the estimated parameter vector of all joints, respectively. Detailed definitions of $\mathbf{Y}^j$ and $\boldsymbol{\theta}^j$ with the Stribeck friction model are provided in Section 4.1.

### 11.3.2 Adaptation Law

To update the estimation of $\boldsymbol{\theta}$, the following update law is proposed, which operates in both the time and iteration domain and is of the same form as in [29]:

$$\text{time-domain update: } \dot{\hat{\boldsymbol{\theta}}}^j = -\boldsymbol{\Gamma}{\mathbf{Y}^j}^T \mathbf{z}^j, \tag{11.9a}$$

$$\text{iteration-domain update: } \hat{\boldsymbol{\theta}}^{j+1}(0) = \hat{\boldsymbol{\theta}}^j(T). \tag{11.9b}$$

$\mathbf{\Gamma}$ is a constant, positive-definite learning matrix, and T denotes the duration of an iteration. $\mathbf{z}_j$ is defined in (11.5), i.e., $\mathbf{z}_j{=}\dot{\mathbf{e}}_j{+}\alpha\mathbf{e}_j$. The time-domain update in (11.9a) is a gradient-type update law as in [32]. The iteration-domain update in (11.9b) means the initial parameter estimate of the next iteration is set as the final parameter estimate of the previous iteration such that parameter estimates remains continuous across iterations.

### 11.3.3 Stability and convergence analysis

In this section, we will prove the stability and error convergence of the proposed control scheme as given by (11.4)-(11.9). We introduce the following two assumptions:

- A1: The iteratively repetitive disturbance input $\mathbf{d}$ in (1) is bounded for all $t \in [0, T]$;
- A2: The resetting condition holds, i.e., $\mathbf{e}^j(0) = \dot{\mathbf{e}}^j(0) = \mathbf{0}; \forall j$.

Under these two assumptions, we have

$$\begin{aligned}&1) \lim_{j\to\infty}\int_0^T \mathbf{e}^j dt = \lim_{j\to\infty}\int_0^T \dot{\mathbf{e}}^j dt = 0,\\ &2) \lim_{j\to\infty}\tilde{\boldsymbol{\theta}}^j \text{ exists and is bounded,}\end{aligned}$$

where $\tilde{\boldsymbol{\theta}} \triangleq \boldsymbol{\theta} - \hat{\boldsymbol{\theta}}^j$ represents the error associated with a mismatch in parameter estimates from their true values.

In order to prove the validity of this theorem, we first consider the application of all control inputs to rearrange the dynamic model of our system. Combining (11.1) with (11.4, 11.5, 11.8) results in the following,

$$\mathbf{M}(\mathbf{q}^j)\ddot{\mathbf{q}}^j{+}\mathbf{C}(\mathbf{q}^j,\dot{\mathbf{q}}^j)\dot{\mathbf{q}}^j{+}\mathbf{G}(\dot{\mathbf{q}}^j){+}\mathbf{d}{=}\mathbf{L}\mathbf{z}^j{+}\mathbf{Y}^j\hat{\boldsymbol{\theta}}^j{+}\mathbf{u}_l^j. \tag{11.10}$$

Notice then that

$$\begin{aligned}\mathbf{M}(\mathbf{q}^j)\dot{\mathbf{z}}^j &= \mathbf{M}(\mathbf{q}^j)\ddot{\mathbf{q}}_d + \alpha\mathbf{M}(\mathbf{q}^j)\dot{\mathbf{e}}^j - \mathbf{M}(\mathbf{q}^j)\ddot{\mathbf{q}}^j,\\ \mathbf{C}(\mathbf{q}^j,\dot{\mathbf{q}}^j)\mathbf{z}^j &= \mathbf{C}(\mathbf{q}^j,\dot{\mathbf{q}}^j)\dot{\mathbf{q}}_d + \alpha\mathbf{C}(\mathbf{q}^j,\dot{\mathbf{q}}^j)\mathbf{e}^j - \mathbf{C}(\mathbf{q}^j,\dot{\mathbf{q}}^j)\dot{\mathbf{q}}^j,\\ \mathbf{Y}^j\boldsymbol{\theta}^j &= \mathbf{M}(\mathbf{q}^j)\ddot{\mathbf{q}}_d + \mathbf{C}(\mathbf{q}^j,\dot{\mathbf{q}}^j)\dot{\mathbf{q}}_d + \mathbf{G}(\dot{\mathbf{q}}^j) + \alpha(\mathbf{M}(\mathbf{q}^j)\dot{\mathbf{e}}^j + \mathbf{C}(\mathbf{q}^j,\dot{\mathbf{q}}^j)\mathbf{e}^j).\end{aligned}$$

Rearranging these three equations, we can rewrite the input to the system in a different form,

$$\mathbf{M}(\mathbf{q}^j)\ddot{\mathbf{q}}^j + \mathbf{C}(\mathbf{q}^j,\dot{\mathbf{q}}^j)\dot{\mathbf{q}}^j + \mathbf{G}(\dot{\mathbf{q}}^j) + \mathbf{d} = \mathbf{Y}^j\boldsymbol{\theta}^j - \mathbf{M}(\mathbf{q}^j)\dot{\mathbf{z}}^j - \mathbf{C}(\mathbf{q}^j,\dot{\mathbf{q}}^j)\mathbf{z}^j + \mathbf{d}. \tag{11.11}$$

If we now define the following term $\tilde{\mathbf{u}}_l^j \triangleq \mathbf{d} - \mathbf{u}_l^j$ and equate the right-hand sides of (11.10) and (11.11), we arrive at the following,

$$\mathbf{M}(\mathbf{q}^j)\dot{\mathbf{z}}^j + \mathbf{C}(\mathbf{q}^j,\dot{\mathbf{q}}^j)\mathbf{z}^j + \mathbf{L}\mathbf{z}^j = \mathbf{Y}^j\tilde{\boldsymbol{\theta}}^j + \tilde{\mathbf{u}}_l^j. \tag{11.12}$$

The term $\tilde{\mathbf{u}}_l^j$ represents error due to the repetitive disturbance in the presence of an iterative input. Assuming that $\mathbf{d}$ is bounded (A1), we can see that the goal of the iterative input is to minimize $\tilde{\mathbf{u}}_l^j$.

The remainder of this proof comes in two parts. First, we show that both $\tilde{\boldsymbol{\theta}}^j$ and $\boldsymbol{z}^j$ remain bounded in the time domain. Second, we will show that the bound on $\tilde{\boldsymbol{\theta}}^j$ will either remain the same or decrease in each subsequent iteration and that $\int_0^T \boldsymbol{z}^{j^T}\boldsymbol{z}^j dt$ will converge to zero.

**Boundedness of Error Signals**

Errors in the time domain will be bounded if inputs to the system are bounded.

*Proof.* Consider, the following Lyapunov candidate function in the time domain,

$$W^j = \frac{1}{2}\boldsymbol{z}^{j^T}\mathbf{M}(\mathbf{q}^j)\boldsymbol{z}^j + \frac{1}{2}\tilde{\boldsymbol{\theta}}^{j^T}\boldsymbol{\Gamma}^{-1}\tilde{\boldsymbol{\theta}}^j. \tag{11.13}$$

Taking its time derivative results in (11.14). Here, note that $\mathbf{M}(\mathbf{q}^j) \triangleq \mathbf{M}^j$, and $\mathbf{C}(\mathbf{q}^j, \dot{\mathbf{q}}^j) \triangleq \mathbf{C}^j$ for brevity,

$$\begin{aligned}
\dot{W}^j &= \boldsymbol{z}^{j^T}\mathbf{M}^j\dot{\boldsymbol{z}}^j + \frac{1}{2}\boldsymbol{z}^{j^T}\dot{\mathbf{M}}^j\boldsymbol{z}^j + \tilde{\boldsymbol{\theta}}^{j^T}\boldsymbol{\Gamma}^{-1}\dot{\tilde{\boldsymbol{\theta}}}^j \\
&= \boldsymbol{z}^{j^T}(\mathbf{Y}^j\tilde{\boldsymbol{\theta}}^j + \tilde{\mathbf{u}}_l^j - \mathbf{C}^j\boldsymbol{z}^j - \mathbf{L}\boldsymbol{z}^j) + \frac{1}{2}\boldsymbol{z}^{j^T}\dot{\mathbf{M}}^j\boldsymbol{z}^j + \tilde{\boldsymbol{\theta}}^{j^T}\boldsymbol{\Gamma}^{-1}\dot{\tilde{\boldsymbol{\theta}}}^j \\
&= \frac{1}{2}\boldsymbol{z}^{j^T}(\dot{\mathbf{M}}^j - 2\mathbf{C}^j)\boldsymbol{z}^j + \boldsymbol{z}^{j^T}\mathbf{Y}^j\tilde{\boldsymbol{\theta}}^j + \tilde{\boldsymbol{\theta}}^{j^T}\boldsymbol{\Gamma}^{-1}\dot{\tilde{\boldsymbol{\theta}}}^j - \boldsymbol{z}^{j^T}\mathbf{L}\boldsymbol{z}^j + \boldsymbol{z}^{j^T}\tilde{\mathbf{u}}_l^j \\
&= -\boldsymbol{z}^{j^T}\mathbf{L}\boldsymbol{z}^j + \boldsymbol{z}^{j^T}\tilde{\mathbf{u}}_l^j \leqslant \|\boldsymbol{z}^j\|(\,-\lambda_{min}(\mathbf{L})\|\boldsymbol{z}^j\| + \|\tilde{\mathbf{u}}_l^j\|\,)
\end{aligned} \tag{11.14}$$

where $\|\bullet\|$ denotes the 2-norm of a vector and $\lambda_{min}(\bullet)$ the minimum eigenvalue of a matrix. We've taken advantage of the fact that $\dot{\mathbf{M}}^j - 2\mathbf{C}^j$ is a skew symmetric matrix (a general property of the robot manipulators, [81], to conclude that the term $\frac{1}{2}\boldsymbol{z}^{j^T}(\dot{\mathbf{M}} - 2\mathbf{C}^j)\boldsymbol{z}^j = 0$. Moreover, $(\boldsymbol{z}^{j^T}\mathbf{Y}^j)\tilde{\boldsymbol{\theta}}^j + \tilde{\boldsymbol{\theta}}^{j^T}\boldsymbol{\Gamma}^{-1}\dot{\tilde{\boldsymbol{\theta}}}^j = 0$ due to (11.9a).

Given this result, we can now establish the passivity of the system through the following proof by contradiction: Assume $\boldsymbol{z}^j(t)$ is not bounded, such that $\lim_{t\to\infty}\|\boldsymbol{z}^j(t)\| = \infty$, and that the time interval $[0,T]$ is sufficiently long enough such that $\boldsymbol{z}^j(T)$ is very large. Then there exists a time,

$$t_b \triangleq \sup\{t \mid 0 < t \leqslant T \text{ and } \|\boldsymbol{z}^j(t)\| \leqslant \frac{\|\tilde{\mathbf{u}}_l^j\|}{\lambda_{min}(\mathbf{L})}\}$$

$$\|\tilde{\boldsymbol{\theta}}^j(t_b)\| = \|\tilde{\boldsymbol{\theta}}^j(0) - \boldsymbol{\Gamma}\int_0^{t_b}\mathbf{Y}^{j^{*T}}\boldsymbol{z}^j(t)d\tau\| \leqslant \|\tilde{\boldsymbol{\theta}}^j(0)\| + \lambda_{max}(\boldsymbol{\Gamma})\sup_{t\in[0,t_b]}(\lambda_{max}(\mathbf{Y}^j))\int_0^{t_b}\|\boldsymbol{z}^j(t)\|d\tau$$

□

Here, we assume $||\tilde{\boldsymbol{\theta}}^{j}(0)||<\infty$. Since both $||\boldsymbol{z}^{j}(t_b)||$ and $||\tilde{\boldsymbol{\theta}}^{j}(t_b)||$ may be large but finite values, the function $W^{j}(t_b)$ is also finite. Additionally, since $||\boldsymbol{z}^{j}|| > \frac{||\tilde{\mathbf{u}}_l^{j}||}{\lambda_{min}(\mathbf{L})}$ $\forall t \in [t_b, T]$, it is also the case that $\dot{W}^{j}(t) < 0$, and accordingly $W^{j}(t) \leqslant W^{j}(t_b) < \infty$ $\forall t \in [t_b, T]$. However, it is not possible that $W^{j}(t)$ be bounded and $||\boldsymbol{z}^{j}(t)||$ not be bounded, and this creates a contradiction. As a result, $||\boldsymbol{z}^{j}||$ must be bounded when $||\tilde{\mathbf{u}}_l^{j}||$ is bounded.

## Convergence in the Iteration Domain

We now analyze how $\tilde{\mathbf{u}}_l^{j}$ changes as j increases, and prove the error $\boldsymbol{z}^{j}$ converges as $j \to \infty$.

*Proof.* Consider the following discrete-time Lyapunov candidate function,

$$V^{j}(t)=\int_{\tau=0}^{T}(\tilde{\mathbf{u}}_l^{j^T}\mathbf{L}^{-1}\tilde{\mathbf{u}}_l^{j})d\tau+\beta\tilde{\boldsymbol{\theta}}^{j^T}(0)\boldsymbol{\Gamma}^{-1}\tilde{\boldsymbol{\theta}}^{j}(0)=V_1^{j}+V_2^{j}. \tag{11.15}$$

where $V_1^{j}$ and $V_2^{j}$ equate to the first and second term of $V^{j}(t)$, respectively. We now define the terms $\Delta\tilde{\mathbf{u}}_l^{j} \triangleq \tilde{\mathbf{u}}_l^{j+1} - \tilde{\mathbf{u}}_l^{j} = (\mathbf{d} - \mathbf{u}_l^{j} - \beta\mathbf{L}\boldsymbol{z}^{j}) - (\mathbf{d} - \mathbf{u}_l^{j}) = -\beta\mathbf{L}\boldsymbol{z}^{j}$ and $\Delta V^{j} \triangleq V^{j+1} - V^{j}$. We then compute the following differences,

$$\begin{aligned}
\Delta V_1^{j} &= \int_0^T \tilde{\mathbf{u}}_l^{j+1^T}\mathbf{L}^{-1}\tilde{\mathbf{u}}_l^{j+1} - \tilde{\mathbf{u}}_l^{j^T}\mathbf{L}^{-1}\tilde{\mathbf{u}}_l^{j}d\tau \\
&= \int_0^T (\Delta\tilde{\mathbf{u}}_l^{j} + \tilde{\mathbf{u}}_l^{j})^T\mathbf{L}^{-1}(\Delta\tilde{\mathbf{u}}_l^{j} + \tilde{\mathbf{u}}_l^{j}) - \tilde{\mathbf{u}}_l^{j^T}\mathbf{L}^{-1}\tilde{\mathbf{u}}_l^{j}d\tau \\
&= \int_0^T \Delta\tilde{\mathbf{u}}_l^{j^T}\mathbf{L}^{-1}\Delta\tilde{\mathbf{u}}_l^{j} + 2\Delta\tilde{\mathbf{u}}_l^{j^T}\mathbf{L}^{-1}\tilde{\mathbf{u}}_l^{j} \\
&= \int_0^T \beta^2\boldsymbol{z}^{j^T}\mathbf{L}\boldsymbol{z}^{j} - 2\beta(\dot{W}^{j} + \boldsymbol{z}^{j^T}\mathbf{L}\boldsymbol{z}^{j})d\tau \\
\Delta V_2^{j} &= \beta(\tilde{\boldsymbol{\theta}}^{j+1^T}(0)\boldsymbol{\Gamma}^{-1}\tilde{\boldsymbol{\theta}}^{j+1}(0) - \tilde{\boldsymbol{\theta}}^{j^T}(0)\boldsymbol{\Gamma}^{-1}\tilde{\boldsymbol{\theta}}^{j}(0))
\end{aligned}$$

From (11.13), it can seen that

$$\begin{aligned}
\int_0^T \dot{W}^{j}d\tau =& \frac{1}{2}\boldsymbol{z}^{j^T}\mathbf{M}(\mathbf{q}^{j})\boldsymbol{z}^{j}\Big|_0^T + \frac{1}{2}\left(\tilde{\boldsymbol{\theta}}^{j^T}(T)\boldsymbol{\Gamma}^{-1}\tilde{\boldsymbol{\theta}}^{j}(T) - \tilde{\boldsymbol{\theta}}^{j^T}(0)\boldsymbol{\Gamma}^{-1}\tilde{\boldsymbol{\theta}}^{j}(0)\right) \\
=& \frac{1}{2}\boldsymbol{z}^{j^T}(T)\mathbf{M}(\mathbf{q}^{j})\boldsymbol{z}^{j}(T) + \frac{1}{2}\tilde{\boldsymbol{\theta}}^{j+1^T}(0)\boldsymbol{\Gamma}^{-1}\tilde{\boldsymbol{\theta}}^{j+1}(0) - \tilde{\boldsymbol{\theta}}^{j^T}(0)\boldsymbol{\Gamma}^{-1}\tilde{\boldsymbol{\theta}}^{j}(0),
\end{aligned}$$

because it is prescribed that $\boldsymbol{\theta}^{j+1}(0) = \boldsymbol{\theta}^{j}(T)$ and by (A2), the initial errors of each iteration $\boldsymbol{z}^{j}(0) = \mathbf{0}$. When we substitute this result into our equation for $\Delta V_1^{j}$ and add $\Delta V_1^{j} + \Delta V_2^{j} = \Delta V^{j}$. The following equation follows,

$$\Delta V^{j}=-\beta\boldsymbol{z}^{j^T}(T)\mathbf{M}(\mathbf{q}^{j})\boldsymbol{z}^{j}(T)-\beta(2-\beta)\int_0^T \boldsymbol{z}^{j^T}\mathbf{L}\boldsymbol{z}^{j}d\tau. \tag{11.16}$$

Due to the form of $\Delta V^j$, the value of $V^j$ will continue to decrease with each subsequent iteration. Since $V^1$ is positive, and $V^j$ is bounded below by zero, then it must converge to some fixed value, $\lim_{j\to\infty} V^j \leqslant V^1$. Consequently, its terms $\tilde{\mathbf{u}}^j$ and $\tilde{\boldsymbol{\theta}}^j$ will also approach zero as $j \to \infty$ and converge to some fixed value. Additionally, since $V^j$ must converge, then $\Delta V^j$ must eventually equal zero. In order for this to occur, it must be true that $\int_0^T \mathbf{z}^{j^T} \mathbf{L} \mathbf{z}^j d\tau = 0$ as $j \to \infty$.

Lastly, if we consider the relationship between $\mathbf{z}^j$ and $\mathbf{e}^j$, we can establish the transfer function $\frac{E^j(s)}{Z^j(s)} = \frac{1}{s+a}$ under (A2). Hence, as long as $a > 0$, it is guaranteed that $\mathbf{z}^j$=$\mathbf{0}$ leads to $\mathbf{e}^j$=$\mathbf{0}$, which further indicates that $\dot{\mathbf{e}}^j = \mathbf{0}$. □

## 11.4 Simulation

### 11.4.1 Manipulator Configuration

We study by simulating an application of the proposed adaptive ILC algorithm to a 3-DOF planar manipulator with joint friction, as shown in Fig. 11.2(a). Due to the difficulty of directly identifying all unknown parameters, we first identify $\nu_{st_i}$ and $\nu_{coul_i}$ offline. Hence, the unknown friction force for each joint $i$ can be represented as

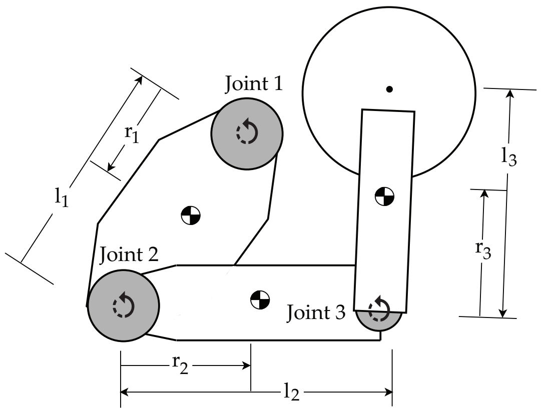
$$\begin{aligned} F_{fri}(q_i, \dot{q}_i) &= \mathbf{Y}_i \boldsymbol{\theta}_i, \\ \boldsymbol{\theta}_i &= [\sqrt{2e}(F_{brk,i} - F_{c,i}) \; F_{c,i} \; f_i]^T \triangleq [b_{i1} \; b_{i2} \; b_{i3}]^T, \\ \mathbf{Y}_i &= [\frac{\dot{q}_i}{\nu_{st}} e^{-\frac{\dot{q}_i}{\nu_{st}}^2} \; \tanh(\frac{\dot{q}_i}{\nu_c}) \; \dot{q}_i]. \end{aligned}$$

Note that $\mathbf{Y}_i$ is the $i$-th row of the regression matrix $\mathbf{Y}$. For the system of interest, the vector $\hat{\boldsymbol{\theta}}^j$ becomes a vertical concatenation of $\hat{\boldsymbol{\theta}}_1^j, \hat{\boldsymbol{\theta}}_2^j, \hat{\boldsymbol{\theta}}_3^j$ and $\mathbf{Y}^j$ becomes a matrix, the rows of which are $\mathbf{Y}_1^j, \mathbf{Y}_2^j, \mathbf{Y}_3^j$. The sampling time is $1ms$.

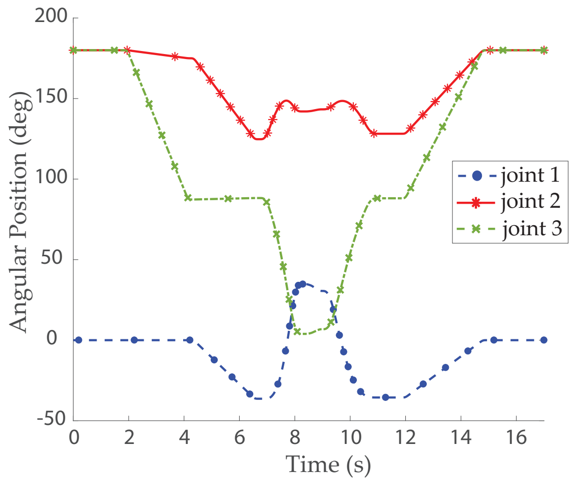
### 11.4.2 Simulation Results

The desired joint trajectories for the manipulator are shown in Fig. 11.2(b). These are typical trajectories for the extension and retraction operations of a silicon wafer handling robot to carry wafers to desired locations. Five iterations of the proposed AILC are run and position errors are shown in Fig. 11.3, demonstrating the effectiveness of the proposed adaptive ILC scheme.

We compare the proposed approach with the one in [29] (noted as "baseline"), with results shown in Fig. 11.4. The proposed algorithm can achieve much better tracking performance than the baseline AILC. Moreover, the torque inputs in Fig. 11.5 show that the adaptive input component contributes less to the overall control effort of the baseline AILC than in the proposed AILC.



**(a)** The 3-DOF manipulator



**(b)** Desired trajectories in joints

**Figure 11.2:** The planar manipulator and desired trajectories in simulation

The performance improvement of the proposed AILC in Fig. 11.4 is a result of the adaptive input design. We can see in (11.7) that the proposed adaptive input accounts for model discrepancies by producing a model-based feedforward input and a feedback compensation signal based on model estimates and tracking error. As the error decreases, the adaptive input gradually reduces to a model-based feedforward control signal. In contrast, the baseline adaptive input in (11.17) also accounts for model discrepancies, but in contrast, its control effort will approach zero as model estimates improve and error decreases.

## 11.5 Experiments on a Wafer Handling Robot Manipulator

### 11.5.1 Experimental Setup

We utilize a 5-DOF Silicon Wafer Handling robot for experimentation. The joints are configured as shown in Fig. 11.6: Joint 1 refers to the base rotation joint, Joint 2 refers to the translation joint in $z$ direction, and Joints 3, 4 and 5 refer to the remaining rotation joints. Each joint can be controlled independently. Among all five joints, Joint 1 has the largest moment of inertia, and Joint 5 suffers from static friction most significantly. These properties make the control of these two joints challenging. Hence, we focused our experiment on Joints 1 and 5. The desired trajectories of these two joints are shown in Fig. 11.6(b).

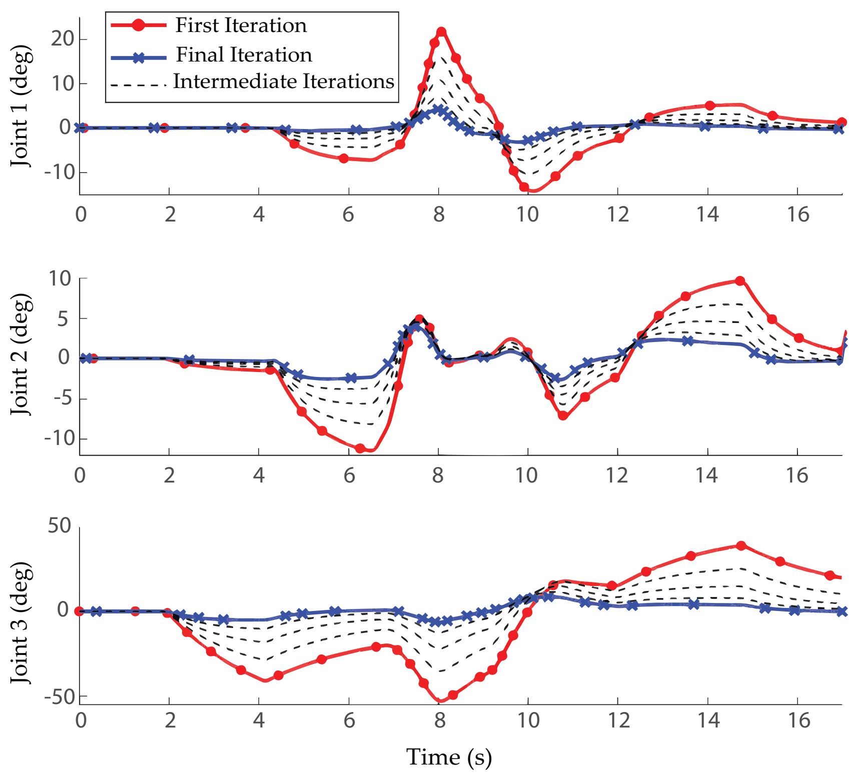


**Figure 11.3:** Position errors of the proposed AILC scheme over five iterations

For both joints, we test both the baseline and proposed AILC approaches. 23 iterations and 13 iterations of the both algorithms were conducted on Joint 1 and Joint 5, respectively. The number of iterations was set such that neither algorithm showed any significant performance improvement after that many iterations.

The baseline adaptive input in [102] and [29] is given by

$$\mathbf{u}_c^j = \hat{\mathbf{M}}_e(\mathbf{q}^j)\ddot{\mathbf{q}}_d + \hat{\mathbf{C}}_e(\mathbf{q}^j, \dot{\mathbf{q}}^j)\dot{\mathbf{q}}_d + \hat{\mathbf{G}}_e(\dot{\mathbf{q}}^j) + \alpha(\hat{\mathbf{M}}(\mathbf{q}^j)\dot{\mathbf{e}}^j + \hat{\mathbf{C}}(\dot{\mathbf{q}}^j)\mathbf{e}^j) \tag{11.17}$$

where $\mathbf{M}_e(\mathbf{q}^j) = \mathbf{M}(\mathbf{q}^j) - \mathbf{M}(\mathbf{q}_d)$ , $\mathbf{C}_e(\mathbf{q}^j, \dot{\mathbf{q}}^j) = \mathbf{C}(\mathbf{q}^j, \dot{\mathbf{q}}^j) - \mathbf{C}(\mathbf{q}_d^j, \dot{\mathbf{q}}_d^j)$, and $\hat{\mathbf{G}}_e(\dot{\mathbf{q}}_j) = \hat{\mathbf{G}}(\dot{\mathbf{q}}^j) - \hat{\mathbf{G}}(\dot{\mathbf{q}}_d^j)$.

## 11.5.2 Experimental Results

Fig. 11.7 gives the position errors in the final iteration and the performance improvements over iterations by using the two AILC schemes. Compared to the baseline, the proposed AILC algorithm results in small performance improvements in Joint 1 but significant improvements in Joint 5, as shown in Fig. 11.7(a). The friction forces in both cases can be better compensated.

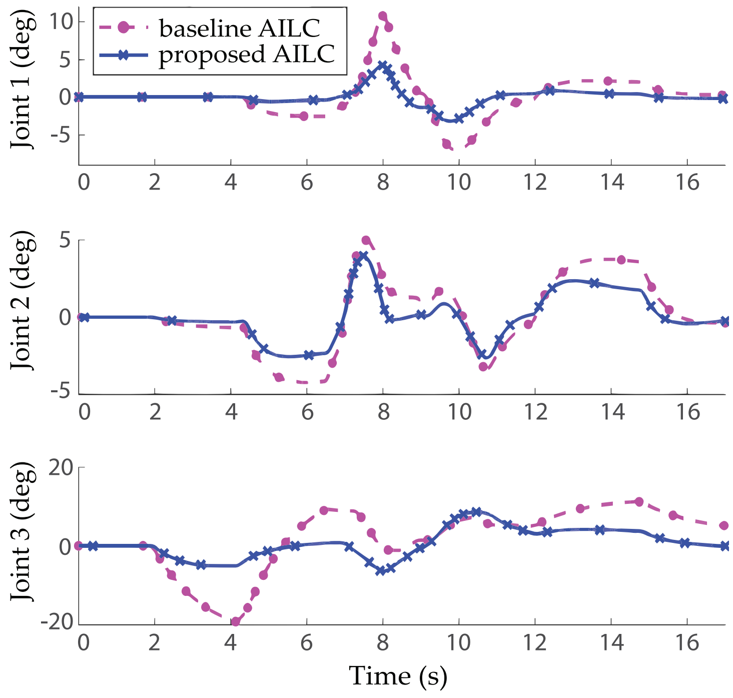


**Figure 11.4:** Comparison result in position errors (final iteration)

As a compact metric of performance, we define the following cost associated with tracking error,

$$J = \int_0^{t_f} \mathbf{z}^T \mathbf{L} \mathbf{z}^j d\tau. \tag{11.18}$$

This metric is a direct result of (11.15) and (11.16), and can be expected converge to zero as $j \to \infty$. Fig. 11.7(b) shows the evolution of J along iterations using the baseline and proposed AILC. We can see that the proposed algorithm leads to great improvements in all iterations of Joint 5 and later iterations of Joint 1.

## 11.6 Chapter Summary

In this chapter, we addressed the fundamental limitation of standard ILC with adaptation. A new, adaptive iterative learning control algorithm was proposed to identify an unknown disturbance model and compensate for it. Proof of error convergence, simulation and experimental studies with the proposed and a baseline AILC were provided with application to robot manipulators subjected to iteration-varying joint friction forces. The results showed that the proposed algorithm performed better than the baseline in terms of reducing tracking performance over iterations while maintaining the same parameter adaptation scheme.

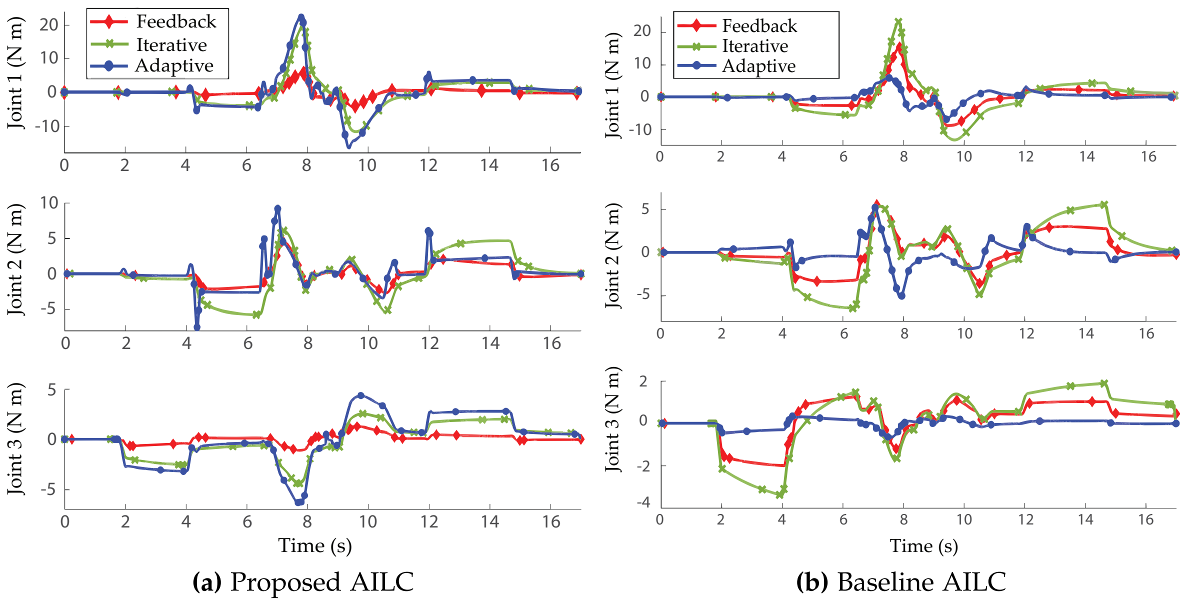
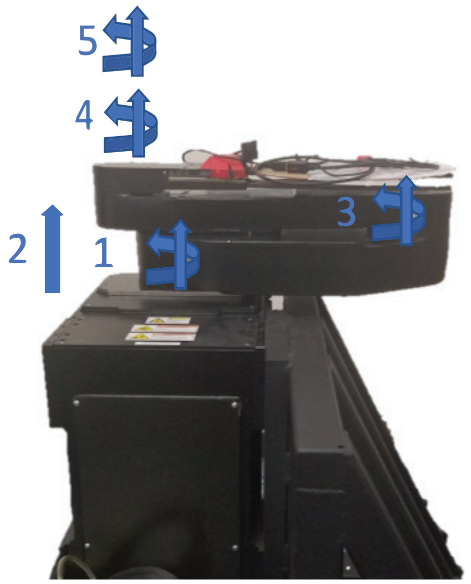


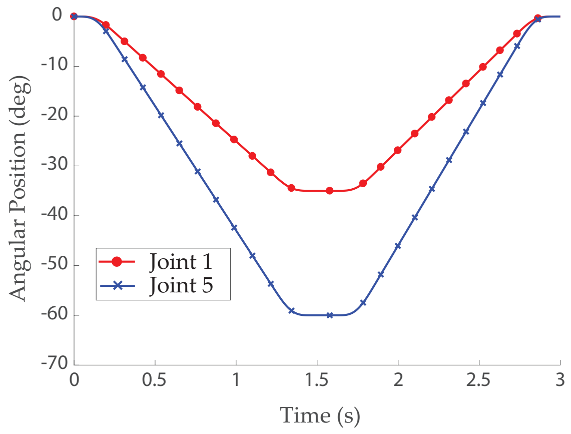
**Figure 11.5:** Torque inputs in the 5th iteration (feedback $\mathbf{u}_f^j$, iterative $\mathbf{u}_l^j$, and adaptive $\mathbf{u}_c^j$)

The proposed approach can be widely applied to autonomous systems subject to iteration-varying friction models. Moreover, estimation of unknown parameters can be extended to include those associated with mass inertia $\mathbf{M}(\mathbf{q})$ and Coriolis $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ terms. By introducing a novel, adaptive input component, Type II non-repetitive disturbances can be better addressed using the proposed approach.

Together with Chapter 10, we have enhanced the performance of traditional ILC with selective learning and adaptability which can handle both external and internal unknown iteration-varying disturbances. Combined with the adaptive feedback controller design in Chapter 9, we have established a two-degree-of-freedom controller design framework to assure that the high-level behaviors can be reliably executed in the presence of a variety of disturbances and uncertainties.

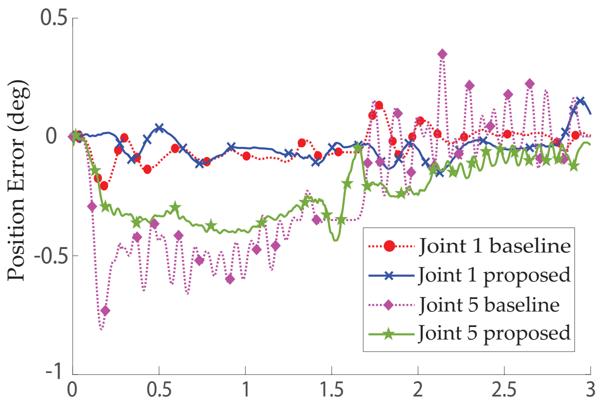


**(a)** The robot (side view)

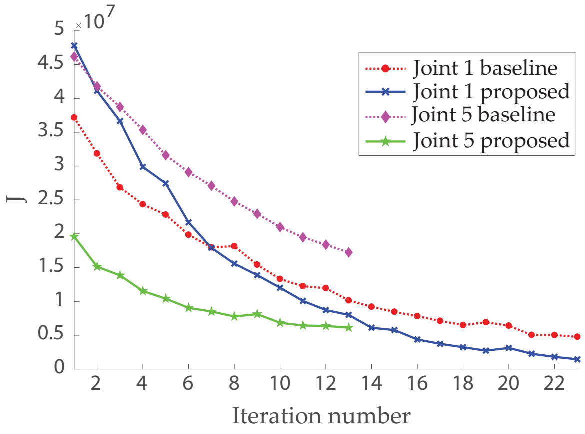


**(b)** Desired trajectories

**Figure 11.6:** The experiment setup and desired trajectories



**(a)** Tracking errors



**(b)** Error evolution

**Figure 11.7:** Tracking errors in final iteration and along iterations

# Chapter 12

# Final Words

We are embracing an era of mixed autonomy. Autonomous systems, such as intelligent robots and driverless vehicles, are expected to enter the social space occupied by humans and intensively interact with them to enable a more efficient society. Compared to traditional automated systems that work in isolated environments with fixed and detailed tasks, autonomous systems need to actively explore efficient ways to achieve more diverse and abstract tasks in a more dynamic environment. Their behavior should be intelligent and of high performance in the presence of uncertainties coming from not only non-intelligent factors such as external vibrations and model uncertainties but also the intelligent behavior of humans which is naturally probabilistic, hierarchical, causal but irrational, socially compliant and time-varying.

This dissertation has focused on the design of such intelligent and high-performance behavior for autonomous systems. It consists of the following two parts: Part I on high-level *hybrid human-machine behavior design* towards seamless and safe interaction with human, and Part II on low-level *individual machine behavior design* to support accurate and reliable execution of the high-level commands. We have leveraged advanced techniques and concepts from control, optimization, learning and cognitive science in both parts.

In Part I, based on a game-theoretic formulation for human-robot interaction, we have represented human behavior via their rewards to optimize (Chapter 2) and discussed three different aspects on the rewards design regarding, respectively, the integration of social compliance into rewards (Chapter 3), the hierarchical structural design and learning in rewards (Chapter 4), and the alternative reward measure based on Cumulative Prospect Theory to describe the irrationality of human behavior (Chapter 5). With such an efficient representation of human behavior, Chapters 6-8 addressed the generation of robotic behavior via the MPC framework in the presence of uncertainties and computation constraints. In Chapter 6, human-like perception behavior, i.e. inferring information from surrounding agents to reduce one's own perception uncertainties, was enabled to tackle perception uncertainties in motion planning. In Chapter 7, an interactive motion planning considering time-varying game policies of human was proposed. Via online estimation of potential game policies that the human might execute, the autonomous systems can generate policy-adaptive behavior to enable smooth interactions. Chapter 8 focused on the efficiency of the behavior generation by leveraging the strength from both optimization and imitation learning. A hierarchical structure with a learning-based long-term policy layer and an optimization based short-term execution layer were combined to enable efficient and safe maneuvers of the robotic systems. In Part II, to assure accurate and reliable execution of the high-level commands, attenuation of external vibrations, particularly the high-frequency ones, has been addressed in Chapter 9. Chapters 10-11 focused on the elimination of repetitive disturbances and model uncertainties via iterative learning control in the presence of non-repetitive components. Selective and adaptive iterative learning control algorithms have been developed and verified.

## 12.1 Open challenges to explore

The work in this dissertation contributes as one step toward safe and efficient mixed autonomy. There are still many directions that the work can be extended in the future to enable seamless human-robot interaction. We list below some of the major directions.

### A unified human behavior model to cover diverse properties

In this dissertation, we have addressed the design of rewards to capture some properties of human behavior such as social compliance, hierarchy, irrationality and time-varyingness. For each of the properties, there are more factors to consider. For instance, how do we systematically capture the difference between single-human behavior and group behavior? How do we extend the hierarchy structure to more layers? And how to summarize and distinguish different categories of individual behavior? Beyond that, there are many more important properties yet to be described and to be considered. For example, due to the computation limit of our brains, human is bounded rational in terms of finding optimal solutions, and autonomous systems need to be aware of that. Human behavior is also emotional. Depending on the mood, human can choose completely different actions given the same scenario. A unified and efficient framework for reward design and learning in the presence of those properties is yet to be explored in the future.

### The ability of life-long learning

Building a human-like autonomous system is the dream of humanity. One of the key features of human as intelligent agents is their life-long learning ability. For instance, human drivers learn basic knowledge for driving during training phase, but they consistently improve their driving skills as they observe more and drive more. As robotic system designers, it is practically not feasible for us to enumerate all possible scenarios and design the best policies for the systems. Hence, autonomous systems should be able to automatically improve and learn. Such life-long learning ability can be summarized into two levels: 1) the ability to identify unsatisfactory performance and to automatically improve that via policy adjustment or knowledge update, and 2) the ability to understand new scenarios, develop new skills, and handle new uncertainties based on previous experience and currently available knowledge. Learning algorithms such as meta-learning is yet to be developed to better support such life-long learning ability.

### Closed-loop system evaluation

Autonomous systems heavily rely on learning or data-driven approaches in both modeling the environment (including human behavior and other influential factors) and generating their own policies. Different from traditional model-based control design which is deterministic, learning-based approaches are mostly probabilistic. Stability analysis techniques for traditional control design are no longer effective under such circumstance and new methodologies are necessary evaluation tools. For instance, for empirical evaluation, we need to construct realistic simulators for autonomous systems in which the simulated scenarios match those in real applications and the other intelligent agents behave like human. We also need to establish a set of metrics to quantify their performance. For theoretic evaluation, as most of the autonomous systems are modularized, and each module generates uncertainties, we need to model the propagation of uncertainties from high-level/upstream modules to low-level/downstream modules, and to the closed-loop performance of the human-robot interactive systems.

## 12.2 Conclusion

As more and more automated systems become autonomous, we are embracing a society with intensive human-robot interaction. Advanced techniques from various scientific fields such as control, optimization, learning and cognitive science may be synergized to conquer more open challenges and eventually achieve safe and efficient mixed autonomy in our society.

# Bibliography

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

[2] Pieter Abbeel and Andrew Y Ng. “Inverse reinforcement learning”. In: *Encyclopedia of machine learning*. Springer, 2011, pp. 554–558.

[3] Najah AbuAli and Hatem Abou-zeid. “Driver Behavior Modeling: Developments and Future Directions”. In: *International Journal of Vehicular Technology* (2016).

[4] Oladapo Afolabi et al. “People as Sensors: Imputing Maps from Human Actions”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Oct. 2018, pp. 2342–2348.

[5] Alexandre Alahi et al. “Social LSTM: Human trajectory prediction in crowded spaces”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 961–971.

[6] Vassili Alexiadis et al. “The Next Generation Simulation Program”. In: *Institute of Transportation Engineers. ITE Journal; Washington* 74.8 (Aug. 2004), pp. 22–26.

[7] Maurice Allais. “Rational man’s behavior in the presence of risk: Critique of the postulates and axioms of the American school”. In: *Econometrica* 21.4 (1953), pp. 503–46.

[8] Samer Ammoun and Fawzi Nashashibi. “Real time trajectory prediction for collision risk estimation between vehicles”. In: *2009 IEEE 5th International Conference on Intelligent Computer Communication and Processing*. IEEE. 2009, pp. 417–422.

[9] Georges S Aoude et al. “Driver Behavior Classification at Intersections and Validation on Large Naturalistic Data Set”. In: *IEEE Transactions on Intelligent Transportation Systems* 13.2 (June 2012), pp. 724–736.

[10] Suguru Arimoto, Sadao Kawamura, and Fumio Miyazaki. “Bettering operation of robots by learning”. In: *Journal of Robotic systems* 1.2 (1984), pp. 123–140.

[11] Brian Armstrong-Hélouvry, Pierre Dupont, and Carlos Canudas De Wit. “A survey of models, analysis tools and compensation methods for the control of machines with friction”. In: *Automatica* 30.7 (1994), pp. 1083–1138.

[12] Takenori Atsumi. “Disturbance suppression beyond Nyquist frequency in hard disk drives”. In: *Mechatronics* 20.1 (2010), pp. 67–73.

[13] M. Bahram et al. “A Game-Theoretic Approach to Replanning-Aware Interactive Scene Prediction and Planning”. In: *IEEE Transactions on Vehicular Technology* 65.6 (June 2016), pp. 3981–3992.

[14] Chris L Baker and Joshua B Tenenbaum. “Modeling human plan recognition using Bayesian theory of mind”. In: *Plan, activity, and intent recognition: Theory and practice* (2014), pp. 177–204.

[15] Tomas Berglund et al. “Planning smooth and obstacle-avoiding B-spline paths for autonomous mining vehicles”. In: *IEEE Transactions on Automation Science and Engineering* 7.1 (2010), pp. 167–172.

[16] Christopher M Bishop. “Mixture density networks”. In: (1994).

[17] Mariusz Bojarski et al. “End to end learning for self-driving cars”. In: *arXiv preprint arXiv:1604.07316* (2016).

[18] Paola Bondi, Giuseppe Casalino, and Lucia Gambardella. “On the iterative learning control theory for robotic manipulators”. In: *IEEE Journal on Robotics and Automation* 4.1 (1988), pp. 14–22.

[19] Francesco Borrelli. “Model Predictive Control for Linear and Hybrid Systems Multiparametric Programming, Lecture notes, Dept. of Mech”. In: *Eng., UCB* (2011).

[20] Maxime Bouton et al. “Scalable Decision Making with Sensor Occlusions for Autonomous Driving”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. May 2018, pp. 2076–2081.

[21] Glenn W Brier. “Verification of forecasts expressed in terms of probability”. In: *Monthey Weather Review* 78.1 (1950), pp. 1–3.

[22] Douglas A Bristow, Marina Tharayil, and Andrew G Alleyne. “A survey of iterative learning control”. In: *IEEE control systems magazine* 26.3 (2006), pp. 96–114.

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

[24] Martin Buehler, Karl Iagnemma, and Sanjiv Singh. *The DARPA urban challenge: autonomous vehicles in city traffic*. Vol. 56. springer, 2009.

[25] Chenyi Chen et al. “Deepdriving: Learning affordance for direct perception in autonomous driving”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 2722–2730.

[26] Xu Chen and M. Tomizuka. “A Minimum Parameter Adaptive Approach for Rejecting Multiple Narrow-Band Disturbances With Application to Hard Disk Drives”. In: *Control Systems Technology, IEEE Transactions on* 20.2 (2012), pp. 408–415.

[27] Xu Chen and Hui Xiao. "Multirate forward-model disturbance observer for feedback regulation beyond Nyquist frequency". In: *Systems & Control Letters* 94 (2016), pp. 181–188.

[28] YangQuan Chen and Kevin L Moore. "Harnessing the nonrepetitiveness in iterative learning control". In: *Proceedings of the 41st IEEE Conference on Decision and Control, 2002.* Vol. 3. IEEE. 2002, pp. 3350–3355.

[29] J. Y. Choi and J. S. Lee. "Adaptive iterative learning control of uncertain robotic systems". In: *IEE Proceedings - Control Theory and Applications* 147.2 (2000), pp. 217–223.

[30] C. Dong, J. M. Dolan, and B. Litkouhi. "Intention estimation for ramp merging control in autonomous driving". In: *2017 IEEE Intelligent Vehicles Symposium (IV).* June 2017, pp. 1584–1589.

[31] Katherine Driggs-Campbell, Vijay Govindarajan, and Ruzena Bajcsy. "Integrating Intuitive Driver Models in Autonomous Planning for Interactive Maneuvers". In: *IEEE Transactions on Intelligent Transportation Systems* 18.12 (Dec. 2017), pp. 3461–3472.

[32] M Feemster et al. "Adaptive control techniques forfrictioncompensation". In: *Mechatronics* 9.2 (1999), pp. 125–145.

[33] Dave Ferguson, Thomas M Howard, and Maxim Likhachev. "Motion planning in urban environments". In: *Journal of Field Robotics* 25.11-12 (2008), pp. 939–960.

[34] Jaime F Fisac et al. "Hierarchical game-theoretic planning for autonomous vehicles". In: (2019), pp. 9590–9596.

[35] Bruce A Francis and Walter Murray Wonham. "The internal model principle of control theory". In: *Automatica* 12.5 (1976), pp. 457–465.

[36] Hiroshi Fujimoto, Fumihiro Kawakami, and Seiji Kondo. "Multirate repetitive control and applications". In: *American Control Conference, 2003. Proceedings of the 2003*. Vol. 4. 2003, pp. 2875–2880.

[37] Enric Galceran, Edwin Olson, and Ryan M Eustice. "Augmented vehicle tracking under occlusions for decision-making in autonomous driving". In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).* Sept. 2015, pp. 3559–3565.

[38] Xinli Geng et al. "A Scenario-Adaptive Driving Behavior Prediction Approach to Urban Autonomous Driving". In: *Applied Sciences* 7.4 (Apr. 2017), p. 426.

[39] Samuel J Gershman, Eric J Horvitz, and Joshua B Tenenbaum. "Computational rationality: A converging paradigm for intelligence in brains, minds, and machines". In: *Science* 349.6245 (2015), pp. 273–278.

[40] David González et al. "A review of motion planning techniques for automated vehicles". In: *IEEE Transactions on Intelligent Transportation Systems* 17.4 (2016), pp. 1135–1145.

[41] Shixiang Gu et al. "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates". In: *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE. 2017, pp. 3389–3396.

[42] Agrim Gupta et al. "Social gan: Socially acceptable trajectories with generative adversarial networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 2255–2264.

[43] M. Hirata. *NSS benchmark problem of hard disk drive system*. http://mizugaki.iis.u-tokyo.ac.jp/nss/. 2007.

[44] Stefan Hoermann et al. "Entering crossroads with blind corners. A safe strategy for autonomous vehicles". In: *Intelligent Vehicles Symposium (IV), 2017 IEEE*. IEEE, 2017, pp. 727–732.

[45] Chun-Te Hsu, Chiang-Ju Chien, and Chia-Yu Yao. "A new algorithm of adaptive iterative learning control for uncertain robotic systems". In: *2003 IEEE International Conference on Robotics and Automation*. Vol. 3. 2003, pp. 4130–4135.

[46] *https://julialang.org*.

[47] Yeping Hu, Liting Sun, and Masayoshi Tomizuka. "Generic Prediction Architecture Considering both Rational and Irrational Driving Behavior". In: *Proceedings of the IEEE Transportation System Conference (ITSC2019)*. 2019.

[48] Yeping Hu, Wei Zhan, and Masayoshi Tomizuka. "A Framework for Probabilistic Generic Traffic Scene Prediction". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. Nov. 2018, pp. 2790–2796.

[49] Yeping Hu, Wei Zhan, and Masayoshi Tomizuka. "Probabilistic Prediction of Vehicle Semantic Intention and Motion". In: *2018 IEEE Intelligent Vehicles Symposium (IV)*. June 2018, pp. 307–313.

[50] Yeping Hu et al. "Multi-Modal Probabilistic Prediction of Interactive Behavior Via an Interpretable Model". In: *Proceedings of the IEEE Intelligent Vehicle Symposium (IV2019)*. 2019.

[51] Constantin Hubmann et al. "A Belief State Planner for Interactive Merge Maneuvers in Congested Traffic". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. Nov. 2018, pp. 1617–1624.

[52] Constantin Hubmann et al. "Automated Driving in Uncertain Environments: Planning With Interaction and Uncertain Maneuver Prediction". In: *IEEE Transactions on Intelligent Vehicles* 3.1 (2018), pp. 5–17.

[53] Joshua Joseph et al. "A Bayesian nonparametric approach to modeling motion patterns". In: *Autonomous Robots* 31.4 (2011), p. 383.

[54] Kosta Jovanovic et al. “Editorial Human-Like Advances in Robotics: Human-Like Advances in Robotics: Motion, Actuation, Sensing, Cognition, and Control”. In: *Frontiers in neurorobotics* 13 (2019), p. 85.

[55] Daniel Kahneman. “Prospect theory: An analysis of decisions under risk”. In: *Econometrica* 47 (1979), p. 278.

[56] Rudolf Emil Kalman. “When is a linear control system optimal?” In: *Journal of Basic Engineering* 86.1 (1964), pp. 51–60.

[57] Nitin R Kapania and J Christian Gerdes. “Path tracking of highly dynamic autonomous vehicle trajectories via iterative learning control”. In: *2015 American Control Conference (ACC)*. IEEE. 2015, pp. 2753–2758.

[58] Sertac Karaman and Emilio Frazzoli. “Optimal kinodynamic motion planning using incremental sampling-based methods”. In: *Decision and Control (CDC), 2010 49th IEEE Conference on*. IEEE. 2010, pp. 7681–7687.

[59] Arne Kesting, Martin Treiber, and Dirk Helbing. “Enhanced intelligent driver model to access the impact of driving strategies on traffic capacity”. In: *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 368.1928 (2010), pp. 4585–4605.

[60] Arne Kesting, Martin Treiber, and Dirk Helbing. “General Lane-Changing Model MOBIL for Car-Following Models”. In: *Transportation Research Record* 1999.1 (Jan. 2007), pp. 86–94.

[61] Raymond J. Kiefer, Jeremy Salinger, and John J. Ference. “Status of NHTSA’s Rear-End Crash Prevention Research Program”. In: June 2005.

[62] Henrik Kretzschmar et al. “Socially compliant mobile robot navigation via inverse reinforcement learning”. In: *The International Journal of Robotics Research* 35.11 (2016), pp. 1289–1307.

[63] Tae-yong Kuc and Jin S Lee. “An adaptive learning control of uncertain robotic systems”. In: *[1991] Proceedings of the 30th IEEE Conference on Decision and Control*. 1991, pp. 1206–1211.

[64] Alex Kuefler et al. “Imitating driver behavior with generative adversarial networks”. In: *Intelligent Vehicles Symposium (IV), 2017 IEEE*. IEEE. 2017, pp. 204–211.

[65] Yoshiaki Kuwata et al. “Real-time motion planning with applications to autonomous urban driving”. In: *IEEE Transactions on Control Systems Technology* 17.5 (2009), pp. 1105–1118.

[66] Ioan Doré Landau, Aurelian Constantinescu, and Daniel Rey. “Adaptive narrow band disturbance rejection applied to an active suspension—an internal model principle approach”. In: *Automatica* 41.4 (2005), pp. 563–574.

[67] Nicholas C Landolfi and Anca D Dragan. "Social Cohesion in Autonomous Driving". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Oct. 2018, pp. 8118–8125.

[68] Jay H Lee, Kwang S Lee, and Won C Kim. "Model-based iterative learning control with a quadratic criterion for time-varying linear systems". In: *Automatica* 36.5 (2000), pp. 641–657.

[69] Richard Lee et al. "Adaptive Iterative Learning Control of Robot Manipulators for Friction Compensation". In: *the 8th IFAC Symposium on Mechatronic Systems (MECHATRONICS 2019) and the 11th IFAC Symposium on Nonlinear Control Systems (NOLCOS 2019)*. IFAC. 2019.

[70] Stéphanie Lefèvre, Dizan Vasquez, and Christian Laugier. "A survey on motion prediction and risk assessment for intelligent vehicles". In: *ROBOMECH journal* 1.1 (2014), p. 1.

[71] Stéphanie Lefèvre, Christian Laugier, and Javier Ibañez Guzmán. "Intention-Aware Risk Estimation for General Traffic Situations, and Application to Intersection Safety". In: *Inria Research Report* 8379 (Oct. 2013).

[72] David Lenz et al. "Deep neural networks for Markovian interactive scene prediction in highway scenarios". In: *Intelligent Vehicles Symposium (IV), 2017 IEEE*. IEEE. 2017, pp. 685–692.

[73] Sergey Levine and Vladlen Koltun. "continuous inverse Optimal control with locally optimal examples," in: *the 29th International Conference on Machine Learning (ICML-12)*. 2012.

[74] Sergey Levine and Vladlen Koltun. "Continuous Inverse Optimal Control with Locally Optimal Examples". In: (2012).

[75] Jiachen Li, Wei Zhan, and Masayoshi Tomizuka. "Generic Vehicle Tracking Framework Capable of Handling Occlusions Based on Modified Mixture Particle Filter". In: *2018 IEEE Intelligent Vehicles Symposium (IV)*. June 2018, pp. 936–942.

[76] Jiachen Li et al. "Coordination and Trajectory Prediction for Vehicle Interactions via Bayesian Generative Modeling". In: *IEEE Intelligent Vehicles Symposium*. 2019.

[77] Jiachen Li et al. "Generic Probabilistic Interactive Situation Recognition and Prediction: From Virtual to Real". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. Nov. 2018, pp. 3218–3224.

[78] Nan Li et al. "Game theoretic modeling of driver and vehicle interactions for verification and validation of autonomous vehicle control systems". In: *IEEE Transactions on control systems technology* 26.5 (2017), pp. 1782–1797.

[79] Xiaohui Li et al. "A unified approach to local trajectory planning and control for autonomous driving along a reference path". In: *Mechatronics and Automation (ICMA), 2014 IEEE International Conference on*. IEEE. 2014, pp. 1716–1721.

[80] Zhao Liang, Guoqiang Zheng, and Jishun Li. "Automatic parking path optimization based on bezier curve fitting". In: *Automation and Logistics (ICAL), 2012 IEEE International Conference on*. IEEE. 2012, pp. 583–587.

[81] Hong-Chin Lin, Tsung-Chieh Lin, and KH Yae. "On the skew-symmetric property of the Newton-Euler formulation for open-chain robot manipulators". In: 3 (1995), pp. 2322–2326.

[82] Changliu Liu. "Designing Robot Behavior in Human-Robot Interactions". PhD thesis. University of California, Berkeley, 2017.

[83] Lennart Ljung. *System Identification: Theory for the User*. Prentice-Hall Information and System Sciences Series. Pearson Education Canada, 1987.

[84] Jeannette AM Lorteije et al. "The formation of hierarchical decisions in the visual cortex". In: *Neuron* 87.6 (2015), pp. 1344–1356.

[85] Yuexin Ma et al. "TrafficPredict: Trajectory prediction for heterogeneous traffic-agents". In: *arXiv preprint arXiv:1811.02146* (2018).

[86] David Madås et al. "On path planning methods for automotive collision avoidance". In: *Intelligent Vehicles Symposium (IV), 2013 IEEE*. IEEE. 2013, pp. 931–937.

[87] Mike McDonald, Jianping Wu, and Mark Brackstone. "Development of a Fuzzy Logic based Microscopic Motorway Simulation Model". In: *Proceedings of Conference on Intelligent Transportation Systems*. Boston, MA, USA: IEEE, 1997, pp. 82–87.

[88] Roel Merry, René van de Molengraft, and Maarten Steinbuch. "Iterative learning control with wavelet filtering". In: *International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal* 18.10 (2008), pp. 1052–1071.

[89] Sandipan Mishra, Joshua Coaplen, and Masavoshi Tomizuka. "Precision positioning of wafer scanners segmented iterative learning control for nonrepetitive disturbances [applications of control]". In: *IEEE Control Systems Magazine* 27.4 (2007), pp. 20–25.

[90] Luis Yoichi Morales et al. "Towards Predictive Driving through Blind Intersections". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. Nov. 2018, pp. 716–722.

[91] Brett R Murphy and Ichiro Watanabe. "Digital shaping filters for reducing machine vibration". In: *IEEE Transactions on Robotics and Automation* 8.2 (1992), pp. 285–289.

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

[93] Samyeul Noh. "Decision-Making Framework for Autonomous Driving at Road Intersections: Safeguarding Against Collision, Overly Conservative Behavior, and Violation Vehicles". In: *IEEE Transactions on Industrial Electronics* 66.4 (Apr. 2019), pp. 3275–3286.

[94] Jean-Jacques E. Slotine and Weiping Li. "On the Adaptive Control of Robot Manipulators". In: *The International Journal of Robotics Research* 6.3 (1987), pp. 49–59.

[95] Shengue Yang, Xiaoping Fan, and An Luo. "Adaptive robust iterative learning control for uncertain robotic systems". In: *Proceedings of the 4th World Congress on Intelligent Control and Automation (Cat. No.02EX527)*. Vol. 2. 2002, pp. 964–968.

[96] Kouhei Ohnishi. "A new servo method in mechatronics". In: *Transactions of Japanese Society of Electrical Engineering* 107.D (1987), pp. 83–86.

[97] Piotr F Orzechowski, Annika Meyer, and Martin Lauer. "Tackling Occlusions & Limited Sensor Range with Set-based Safety Verification". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. Nov. 2018, pp. 1729–1736.

[98] Martin J Osborne and Ariel Rubinstein. *A course in game theory*. 1994.

[99] Chris J Ostafew, Angela P Schoellig, and Timothy D Barfoot. "Visual teach and repeat, repeat, repeat: Iterative learning control to improve mobile robot path tracking in challenging outdoor environments". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2013, pp. 176–181.

[100] Chee Khiang Pang, Weili Yan, and Chunling Du. "Multirate identification of mechanical resonances beyond the Nyquist frequency in high-performance mechatronic systems". In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 46.4 (2016), pp. 573–581.

[101] Benjamas Panomruttanarug. "Application of iterative learning control in tracking a Dubin's path in parallel parking". In: *International Journal of Automotive Technology* 18.6 (2017), pp. 1099–1107.

[102] BH Park, Tae-Yong Kuc, and Jin S Lee. "Adaptive learning control of uncertain robotic systems". In: *International Journal of Control* 65.5 (1996), pp. 725–744.

[103] Minh Q Phan and Richard W Longman. "Higher-order iterative learning control by pole placement and noise filtering". In: *IFAC Proceedings Volumes* 35.1 (2002), pp. 25–30.

[104] Derek J. Phillips, Tim A. Wheeler, and Mykel J. Kochenderfer. "Generalizable intention prediction of human drivers at intersections". In: *Intelligent Vehicles Symposium (IV), 2017 IEEE*. IEEE, 2017, pp. 1665–1670.

[105] Fabian Poggenhans et al. "Lanelet2: A high-definition map framework for the future of automated driving". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2018, pp. 1672–1679.

[106] Aris Polychronopoulos et al. "Sensor fusion for predicting vehicles' path for collision avoidance systems". In: *IEEE Transactions on Intelligent Transportation Systems* 8.3 (2007), pp. 549–562.

[107] Dean A Pomerleau. *ALVINN, an autonomous land vehicle in a neural network*. Tech. rep. Carnegie Mellon University, Computer Science Department, 1989.

[108] David Premack and Guy Woodruff. "Does the chimpanzee have a theory of mind?" In: *Behavioral and brain sciences* 1.4 (1978), pp. 515–526.

[109] Braden A Purcell and Roozbeh Kiani. "Hierarchical decision processes that operate over distinct timescales underlie choice and changes in strategy". In: *Proceedings of the national academy of sciences* 113.31 (2016), E4531–E4540.

[110] Jannik Quehl et al. "How Good is My Prediction? Finding a Similarity Measure for Trajectory Prediction Evaluation." In: *2017 IEEE 18th International Conference on Intelligent Transportation Systems (ITSC)*. 2017, pp. 120–125.

[111] Lawrence R Rabiner. "A tutorial on hidden Markov models and selected applications in speech recognition". In: *Proceedings of the IEEE* 77.2 (1989), pp. 257–286.

[112] Mizanur Rahman et al. "Review of Microscopic Lane-Changing Models and Future Research Opportunities". In: *IEEE Transactions on Intelligent Transportation Systems* 14.4 (Dec. 2013), pp. 1942–1956.

[113] Rajesh Rajamani. *Vehicle dynamics and control*. Springer Science & Business Media, 2011.

[114] OF Risk and Daniel Bernoulli. "Exposition of a new theory on the measurement". In: *Econometrica* 22.1 (1954), pp. 23–36.

[115] Stéphane Ross, Geoffrey J Gordon, and Drew Bagnell. "A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning." In: *AISTATS*. Vol. 1. 2. 2011, p. 6.

[116] Iuliana Rotariu, Maarten Steinbuch, and Rogier Ellenbroek. "Adaptive iterative learning control for high precision motion systems". In: *IEEE Transactions on Control Systems Technology* 16.5 (2008), pp. 1075–1082.

[117] Dorsa Sadigh. "Safe and interactive autonomy: control, learning, and verification". PhD thesis. UC Berkeley, 2017.

[118] Dorsa Sadigh et al. "Information Gathering Actions over Human Internal State". In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2016, pp. 66–73.

[119] Dorsa Sadigh et al. "Planning for Autonomous Cars that Leverage Effects on Human Actions." In: *Robotics: Science and Systems*. 2016.

[120] Morteza Sarafyazd and Mehrdad Jazayeri. "Hierarchical reasoning by neural circuits in the frontal cortex". In: *Science* 364.6441 (2019), eaav8911.

[121] Matthias Schreier, Volker Willert, and Jürgen Adamy. "An Integrated Approach to Maneuver-Based Trajectory Prediction and Criticality Assessment in Arbitrary Road Environments". In: *IEEE Transactions on Intelligent Transportation Systems* 17.10 (Oct. 2016), pp. 2751–2766.

[122] Reinhard Selten. "Bounded rationality". In: *Journal of Institutional and Theoretical Economics (JITE)/Zeitschrift für die gesamte Staatswissenschaft* 146.4 (1990), pp. 649–658.

[123] Masamichi Shimosaka et al. "Predicting driving behavior using inverse reinforcement learning with multiple reward functions towards environmental diversity". In: *Intelligent Vehicles Symposium (IV), 2015 IEEE*. IEEE. 2015, pp. 567–572.

[124] Abraham Castellanos Silva, Ioan Doré Landau, and Tudor-Bogdan Airimiţoaie. "Direct adaptive rejection of unknown time-varying narrow band disturbances applied to a benchmark problem". In: *European Journal of Control* 19.4 (2013), pp. 326–336.

[125] David Silver et al. "Mastering the game of Go with deep neural networks and tree search". In: *nature* 529.7587 (2016), p. 484.

[126] Marwaan Simaan and Jose B Cruz. "On the Stackelberg strategy in nonzero-sum games". In: *Journal of Optimization Theory and Applications* 11.5 (1973), pp. 533–555.

[127] Andrew Smyth and Meiliang Wu. "Multi-rate Kalman filtering for the data fusion of displacement and acceleration response measurements in dynamic system monitoring". In: *Mechanical Systems and Signal Processing* 21.2 (2007), pp. 706–723.

[128] Liting Sun, Xu Chen, and Masayoshi Tomizuka. "Adaptive Suppression of High-Frequency Wide-Spectrum Vibrations with Application to Disk Drive Systems". In: *2014 ASME Dynamic Systems and Control (DSC) Conference*. San Antonio, Texas, USA, 2014.

[129] Liting Sun, Xu Chen, and Masayoshi Tomizuka. "Selective iterative learning control with non-repetitive disturbance rejection". In: *Proceedings of 2014 International Symposium on Flexible Automation*. 2014.

[130] Liting Sun and Masayoshi Tomizuka. "Multirate Adaptive Disturbance Suppression Beyond Nyquist Frequency". In: *Proceedings of the International Symposium on Flexible Automation*. The Institute of Systems, Control and Information Engineers. 2018, pp. 262–270.

[131] Liting Sun and Masayoshi Tomizuka. "Multirate Vibration Attenuation Beyond Nyquist Frequency with Performance, Stability and Robustness Analysis". In: *IFAC-PapersOnLine* 50.1 (2017), pp. 10857–10863.

[132] Liting Sun, Wei Zhan, and Masayoshi Tomizuka. "Probabilistic Prediction of Interactive Driving Behavior via Hierarchical Inverse Reinforcement Learning". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. Nov. 2018, pp. 2111–2117.

[133] Liting Sun et al. "A fast integrated planning and control framework for autonomous driving via imitation learning". In: *ASME 2018 Dynamic Systems and Control Conference*. American Society of Mechanical Engineers. 2018.

[134] Liting Sun et al. "Behavior Planning of Autonomous Cars with Social Perception". In: *IEEE Intelligent Vehicles Symposium*. 2019.

[135] Liting Sun et al. "Courteous autonomous cars". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 663–670.

[136] Liting Sun et al. "Interpretable modelling of driving behaviors in interactive driving scenarios based on cumulative prospect theory". In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE. 2019, pp. 4329–4335.

[137] Morikazu Takegaki and Suguru Arimoto. "A new feedback method for dynamic control of manipulators". In: *Journal of Dynamic Systems, Measurement, and Control* 103.2 (1981), pp. 119–125.

[138] Alireza Talebpour, Hani S Mahmassani, and Samer H Hamdar. "Modeling lane-changing behavior in a connected environment: A game theory approach". In: *Transportation Research Procedia* 7 (2015), pp. 420–440.

[139] Abdelhamid Tayebi. "Adaptive iterative learning control for robot manipulators". In: *Automatica* 40.7 (2004), pp. 1195–1203.

[140] *The KITTI Vision Benchmark Suite, 3D object detection*. URL: `http://www.cvlibs.net/datasets/kitti/eval_object.php?obj_benchmark=3d`.

[141] Ran Tian et al. "Adaptive game-theoretic decision making for autonomous vehicle control at roundabouts". In: *2018 IEEE Conference on Decision and Control (CDC)*. IEEE. 2018, pp. 321–326.

[142] Masayoshi Tomizuka. "Multi-rate control for motion control applications". In: *Advanced Motion Control, 2004. AMC '04. The 8th IEEE International Workshop on*. 2004, pp. 21–29.

[143] Masayoshi Tomizuka. "Zero phase error tracking algorithm for digital control". In: (1987).

[144] Quan Tran and Jonas Firl. "Modelling of traffic situations at urban intersections with probabilistic non-parametric regression". In: *2013 IEEE Intelligent Vehicles Symposium (IV)*. June 2013, pp. 334–339.

[145] Martin Treiber, Ansgar Hennecke, and Dirk Helbing. "Congested Traffic States in Empirical Observations and Microscopic Simulations". In: *Physical Review E* 62.2 (Aug. 2000), pp. 1805–1824.

[146] Amos Tversky and Daniel Kahneman. "Advances in prospect theory: Cumulative representation of uncertainty". In: *Journal of Risk and uncertainty* 5.4 (1992), pp. 297–323.

[147] Sarah Vincent and Shaun Gallagher. "Are chimpanzees socially enactive?" In: *The Routledge Handbook of Philosophy of Animal Minds* (2017), p. 280.

[148] Mingyu Wang et al. "Game Theoretic Planning for Self-Driving Cars in Competitive Scenarios". In: *Robotics: Science & Systems*. 2019.

[149] Wenshuo Wang, Junqiang Xi, and Huiyan Chen. "Modeling and Recognizing Driver Behavior Based on Driving Data: A Survey". In: *Mathematical Problems in Engineering* (2014).

[150] Cathy Wu et al. "Emergent Behaviors in Mixed-Autonomy Traffic". In: *Conference on Robot Learning*. 2017, pp. 398–407.

[151] Yutaka Yamamoto, Kaoru Yamamoto, and Masaaki Nagahara. "Tracking of signals beyond the Nyquist frequency". In: *Decision and Control (CDC), 2016 IEEE 55th Conference on*. IEEE. 2016, pp. 4003–4008.

[152] Weili Yan, Chee Khiang Pang, and Chunling Du. "Disturbance observer-based multirate control for rejecting periodic disturbances to the Nyquist frequency and beyond". In: *Automatica* 82 (2017), pp. 49–58.

[153] Je Hong Yoo. "A Game Theory Based Model of Human Driving with Application to Autonomous and Mixed Driving". PhD thesis. 2014.

[154] *A Stackelberg Game Theoretic Driver Model for Merging*. Dynamic Systems and Control Conference. Oct. 2013.

[155] Ming-Yuan Yu, Ram Vasudevan, and Matthew Johnson-Roberson. "Occlusion-Aware Risk Assessment for Autonomous Driving in Urban Environments". In: *arXiv:1809.04629 [cs]* (Sept. 2018). arXiv: 1809.04629.

[156] Shuwen Yu and Masayoshi Tomizuka. "Performance enhancement of iterative learning control system using disturbance observer". In: *2009 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*. IEEE. 2009, pp. 987–992.

[157] Wei Zhan et al. "A non-conservatively defensive strategy for urban autonomous driving". In: *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. Nov. 2016, pp. 459–464.

[158] Wei Zhan et al. "Constructing a Highly Interactive Vehicle Motion Dataset". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2019.

[159] Wei Zhan et al. "INTERACTION Dataset: An INTERnational, Adversarial and Cooperative moTION Dataset in Interactive Driving Scenarios with Semantic Maps". In: *arXiv preprint arXiv:1910.03088* (2019).

[160] Wei Zhan et al. "Spatially-partitioned environmental representation and planning architecture for on-road autonomous driving". In: *Intelligent Vehicles Symposium (IV), 2017 IEEE*. IEEE. 2017, pp. 632–639.

[161] Wei Zhan et al. “Towards a fatality-aware benchmark of probabilistic reaction prediction in highly interactive driving scenarios”. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2018, pp. 3274–3280.

[162] Bin Zhang, Danwei Wang, and Yongqiang Ye. “Wavelet transform-based frequency tuning ILC”. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 35.1 (2005), pp. 107–114.

[163] Chengyuan Zhang et al. “A General Framework of Learning Multi-Vehicle Interaction Patterns from Video”. In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE. 2019, pp. 4323–4328.

[164] Minghui Zheng, Liting Sun, and Masayoshi Tomizuka. “Multi-rate Observer Based Sliding Mode Control with Frequency Shaping for Vibration Suppression Beyond Nyquist Frequency”. In: *Proceedings of 7th IFAC Symposium on Mechatronic Systems, IFAC Mechatronics 2016*. 2016, pp. 320–325.

[165] Minghui Zheng et al. “Design of arbitrary-order robust iterative learning control based on robust control theory”. In: *Mechatronics* 47 (2017), pp. 67–76.

[166] Brian D Ziebart et al. “Maximum Entropy Inverse Reinforcement Learning.” In: *AAAI*. Vol. 8. Chicago, IL, USA. 2008, pp. 1433–1438.

[167] Julius Ziegler et al. “Making Bertha drive—An autonomous journey on a historic route”. In: *IEEE Intelligent Transportation Systems Magazine* 6.2 (2014), pp. 8–20.

[168] Julius Ziegler et al. “Trajectory planning for Bertha - A local, continuous method”. In: *Intelligent Vehicles Symposium Proceedings, 2014 IEEE*. IEEE. 2014, pp. 450–457.

[169] Ariel Zylberberg et al. “Serial, parallel and hierarchical decision making in primates”. In: *eLife* 6 (2017), e17331.