UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Autonomous driving: learning to make decisions in uncertain environments

Júnior Anderson Rodrigues da Silva

Tese de Doutorado do Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional (PPG-CCMC)

ICMC USP SÃO CARLOS

Júnior Anderson Rodrigues da Silva

Autonomous driving: learning to make decisions in uncertain environments

Doctoral dissertation submitted to the Instituto de Ciências Matemáticas e de Computação – ICMC-USP, in partial fulfillment of the requirements for the degree of the Doctorate Program in Computer Science and Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Prof. Dr. Denis Fernando Wolf

USP – São Carlos
August 2023

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

R696a

Rodrigues da Silva, Júnior Anderson
Autonomous driving: learning to make decisions in uncertain environments / Júnior Anderson Rodrigues da Silva; orientador Denis Fernando Wolf. -- São Carlos, 2023.
117 p.

Tese (Doutorado - Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2023.

1. Veículos Autônomos. 2. Tomada de Decisão na Presença de Incertezas. 3. Planejamento de Movimento. 4. Inteligência Artificial. 5. Aprendizado de Máquina. I. Fernando Wolf, Denis, orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Júnior Anderson Rodrigues da Silva

Direção autônoma: apredendo a tomar decisões na presença de incertezas

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Doutor em Ciências – Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientador: Prof. Dr. Denis Fernando Wolf

USP – São Carlos
Agosto de 2023

Este trabalho é dedicado à minha família e
a todos aqueles que persistem em busca de seus sonhos.

ACKNOWLEDGEMENTS

Gostaria de agradecer imensamente ao meu orientador Professor Denis Fernando Wolf e ao meu co-orientador Professor Valdir Grassi Junior por terem aceitado me orientar durante o meu doutorado. Todo o apoio e importantes conselhos que me deram tornaram possível a realização deste trabalho. Sou muito grato ao apoio dado por todos os meus colegas do Laboratório de Robótica Móvel (LRM), em especial meus colegas Mariana, Jean, Tiago, Iago, Luis, Carlos, Angelica, Viviana, Victor e Isadora, que sempre disponibilizaram um pouco do seu tempo para ouvir minhas dúvidas e me aconselharem em momentos difíceis, como também por compartilharem seus momentos felizes. Sentirei muita saudade das festinhas de aniversário! Além disso, a assistência fornecida pelo Instituto de Ciências Matemáticas e de Computação foi essencial para a conclusão deste trabalho. Meu muito obrigado a todos os funcionários do ICMC e de empresas terceiras, responsáveis por toda a parte burocrática do instituto, como também pela manutenção e limpeza dos prédios. Ademais, gostaria de agradecer à cidade de São Carlos por ter me acolhido por todos esses anos.

Agradeço também a todos aqueles que sempre me apoiaram e sempre acreditaram que eu conseguiria chegar aqui. Gostaria de agradecer a todos os professores que passaram pela minha vida, em especial a Edilene Leal (*in memoriam*), que me fez despetar minha paixão pela Matemática e ao Professor Janison Rodrigues, que, além de orientador, se tornou um amigo durante minha graduação. Além disso, sou muito grato a todos os meus amigos que me acompanharam durante toda minha jornada até aqui, em especial a Joison Pereira e Elias Camargos (*in memoriam*).

Este trabalho teria sido impossível sem o apoio de minha mãe Ivone, de meu pai José, de minha irmã Josiane, de minha avó Josina e de minha namorada Natália, que estiveram ao meu lado em todos os desafios até a conclusão desta tese de doutorado.

Eu gostaria de agradecer à FAPESP (processo nº 2018/19732-5, Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP)), à CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Código de Financiamento 001 e processo nº 88887.136349/2017-00), ao CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico - processo nº 465755/2014-3) e ao INCT (Institutos Nacionais de Ciência e Tecnologia) pelo apoio financeiro desta pesquisa e do projeto, sem o qual eu não teria concluído esta tese.

“A resposta certa não importa nada:
o essencial é que as perguntas estejam certas.”
(Mario Quintana)

RESUMO

SILVA, J. A. R. **Direção autônoma: apredendo a tomar decisões na presença de incertezas**. 2023. 117 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2023.

Um veículo que navega em um ambiente urbano deve obedecer às regras de trânsito, definindo corretamente sua velocidade para ficar abaixo do limite de velocidade da estrada e evitar colisões. Este é presumivelmente o cenário que os veículos autônomos enfrentarão: eles compartilharão as vias de tráfego com outros veículos (autônomos ou não), interagindo cooperativamente com eles. Em outras palavras, os veículos autônomos não devem apenas seguir as regras de trânsito, mas também devem se comportar de maneira semelhante a outros veículos. Porém, a especificação manual de tal comportamento é um trabalho demorado e sujeito a erros, visto que dirigir em vias urbanas é uma tarefa complexa, que envolve diversos fatores. Além disso, uma vez que a interação entre os veículos é inerente à condução, inferir o movimento dos veículos ao redor é essencial para proporcionar uma navegação mais fluida, evitando um comportamento excessivamente reativo. Nesse sentido, incertezas provenientes de sensores com algum grau de imprecisão, como também do comportamento desconhecido de outros veículos não podem ser negligenciadas de forma a garantir tomadas de decisão seguras e confiáveis.

Nesta tese, propomos o uso do Processo de Decisão de Markov Parcialmente Observável (POMDP) para resolver o problema de informação incompleta inerente ao planejamento de movimento para veículos autônomos. Também propomos uma variante do Aprendizagem por Reforço Inverso (IRL) baseado no princípio da Entropia Máxima para aprender o comportamento de motoristas humanos a partir de demonstrações. Três diferentes cenários urbanos são abordados ao longo deste trabalho: planejamento longitudinal em cruzamentos com semáforo considerando medições ruidosas de sensores; planejamento longitudinal e lateral em vias de múltiplas faixas na presença de outros veículos, em que a intenção dos mesmos de mudar de faixa é inferida a partir de uma sequência de observações; planejamento longitudinal e lateral durante manobras para adentrar vias movimentadas em um cenário altamente interativo, no qual o comportamento do veículo autônomo é aprendido a partir de dados reais contendo demonstrações humanas. Os resultados mostram que nossos métodos se comparam favoravelmente a abordagens que negligenciam a incerteza durante o planejamento, e também podem melhorar o desempenho do aprendizado por IRL, o que agrega segurança e confiabilidade na tomada de decisão.

Palavras-chave: Veículos Autônomos, Tomada de Decisão na Presença de Incertezas, Planejamento de Movimento, Predição de Movimento, Processo de Decisão de Markov Parcialmente Observável (POMDP), Aprendizagem por Reforço Inverso (IRL), Robótica, Inteligência Artificial, Aprendizado de Máquina.

ABSTRACT

SILVA, J. A. R. **Autonomous driving: learning to make decisions in uncertain environments**. 2023. 117 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2023.

A vehicle navigating in an urban environment must obey traffic rules by properly setting its speed in order to stay below the road speed limit and avoiding collisions. This is presumably the scenario that autonomous vehicles will face: they will share the traffic roads with other vehicles (autonomous or not), cooperatively interacting with them. In other words, autonomous vehicles should not only follow traffic rules, but should also behave in such a way that resembles other vehicles behavior. However, manually specification of such behavior is a time-consuming and error-prone work, since driving in urban roads is a complex task, which involves many factors. Furthermore, since the interaction between vehicles is inherent to driving, inferring surrounding vehicles' motion is essential to provide a more fluid navigation, avoiding a over-reactive behavior. In this sense, the uncertainty coming from noisy sensor measurements and unknown surrounding vehicles behavior cannot been neglected in order to guarantee safe and reliable decisions.

In this thesis, we propose using Partially Observable Markov Decision Process (POMDP) to address the problem of incomplete information inherent of motion planning for autonomous driving. We also propose a variant of Maximum Entropy Inverse Reinforcement Learning (IRL) to learn human expert behavior from demonstration. Three different urban scenarios are covered throughout this work: longitudinal planning at signalized intersection by considering noisy measurements sensor; longitudinal and lateral planning on multi-lane roads in the presence of surrounding vehicles, in which their intention of changing lane are inferred from sequential observations; longitudinal and lateral planning during merge maneuvers in a highly interactive scenario, in which the autonomous vehicle behavior is learned from real data containing human demonstrations. Results show that our methods compare favorably to approaches that neglected uncertainty during planning, and also can improve the IRL performance, which adds safety and reliability in the decision-making.

Keywords: Autonomous Driving, Decision-making under Uncertainty, Motion Planning, Motion Prediction, Partially Observed Markov Decision Process (POMDP), Inverse Reinforcement Learning (IRL), Robotics, Artificial Intelligence, Machine Learning.

LIST OF FIGURES

LIST OF ALGORITHMS

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

AI	Artificial Intelligence
ANN	Artificial Neural Network
CARLA	Car Learning to Act
CNN	Convolutional Neural Network
DBN	Dynamic Bayesian Networks
DCU	Darpa Urban Challenge
DL	Deep Learning
DP	Dynamic Programming
EM	Expectation Maximization
FSM	Finite State Machines
GDP	Gross Domestic Product
GPS	Global Positioning System
GPU	Graphics Processing Unit
HSM	Hierarchical State Machine
IMD	Intelligent Driving Model
IMU	Inertial Measurement Unit
IRL	Inverse Reinforcement Learning
MDP	Markov Decision Process
MOBIL	Minimizing Overall Braking Induced by Lane Change
MPC	Model Predictive Control
POMCP	Partially Observable Monte Carlo Planning
POMDP	Partially Observable Markov Decision Process
RL	Reinforcement Learning
ROS	Robot Operating System
SPaT	Signal Phasing and Timing
SPN	Signal Processing Network
V2V	Vehicle to Vehicle
WHO	World Health Organization

CONTENTS

CHAPTER

1

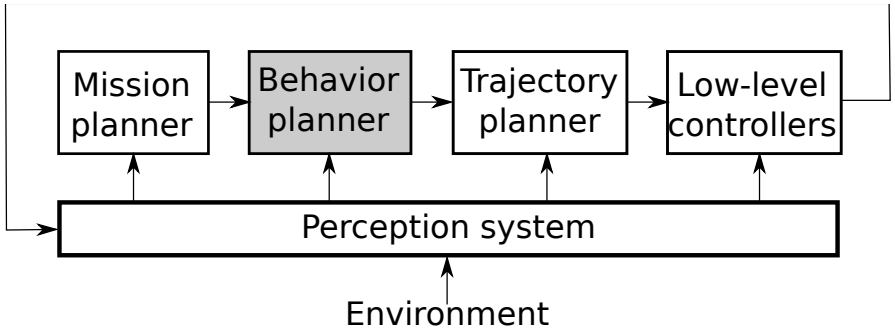
INTRODUCTION

According to the World Health Organization (WHO), 1.2 million people die every year due to traffic accidents, making them the leading cause of death among people from 15 to 29 years old (WHO, 2015). In Brazil, the deaths reach 49,000 per year, making it the fourth country in America in number of traffic accident deaths. In addition, 1.2% of Brazil's Gross Domestic Product (GDP) is lost due to accidents (WHO, 2015). The WHO points out that many accidents are due to improper driving behavior, such as talking on the mobile phone, exceeding the speed limit or being under the influence of alcohol or psychoactive drugs, whether legal or illegal. Therefore, lives could be saved if mistakes of this nature could be avoided. Moreover, the costs arising from these accidents could be invested in other areas, such as education, health and safety.

The number of accidents resulting from human error could be significantly reduced by using robotic cars with systems that allow autonomous navigation, due to their rapid processing and decision making in various situations (LITMAN, 2014). The use of mobile robots to transport people and cargo has been intensively studied in recent years (RANFT; STILLER, 2016; PADEN *et al.*, 2016). Researchers around the world focus their efforts on adapting commercial cars to operate completely autonomously, the so-called *autonomous vehicles*. Autonomous vehicles, also known as driverless cars or self-driving cars, are vehicles that can sense their environment and navigate without human input. They combine a variety of sensors to perceive their surroundings, such as radar, lidar, Global Positioning System (GPS) and computer vision. Advanced control systems interpret sensory information to identify appropriate navigation paths and implement corresponding actions. Benefits of this technology include improved safety due to reduced risk of human error, increased efficiency through optimized routing and decreased pollution from less fuel consumption.

Commonly, the decision-making architecture of autonomous vehicles is divided into simpler components or *layers* (PADEN *et al.*, 2016), as shown in Fig. 1. In the first layer of the decision-making hierarchy , the vehicle needs to find a route considering the connections between the roads that constitute the environment which it is inserted in, the so-called *mission*

Figure 1 – Decision-making hierarchy.



Source: Elaborated by the author.

planning. During navigation, the vehicle must deal with static and dynamic obstacles (other moving vehicles) that can be found along the way, identified through sensors such as cameras, lasers and radars. The presence of dynamic obstacles requires the vehicle to make fundamental navigation decisions. These decisions are performed by the *behavioral planner* in order to minimize the risk of accidents and to accomplish the planned goal, such as overtaking a slower vehicle. Right after deciding on the best strategy, for example changing lanes in the event of an overtaking, it is necessary to calculate a *trajectory*, which can be understood as a reference that will lead to the successful execution of the maneuver. The planned trajectory is executed by the *controller-layer*, which is the lowest layer of the decision-making hierarchy. Sensors such as GPS, Inertial Measurement Unit (IMU), cameras and lasers are used to estimate the vehicle's position. Actuators, such as steering wheel steering, accelerator and brake are used in order to keep the vehicle on the calculated trajectory. Despite the notable improvements in this field, mainly after the Darpa Challenge (BUEHLER; IAGNEMMA; SINGH, 2007) and the subsequent Darpa Urban Challenge (DCU) (URMSON *et al.*, 2008; MONTEMERLO *et al.*, 2008), many challenges still need to be overcome to achieve safe, reliable autonomous systems.

An open problem in autonomous vehicles research is dealing with *uncertainty* while making decisions (SCHWARTING; ALONSO-MORA; RUS, 2018). The main source of uncertanty in autonomous driving field comes from sensor noisy measurements and unknown information, such as occlusions, traffic lights cycle duration, as well as pedestrian, cyclists and surrounding vehicles future trajectories. Neglecting such sources of uncertainty may lead to erroneous planned behaviors or even to unsafe conditions. Besides that, another point of attention of the self driving community is how to define a properly behavior for the controlled vehicle. During navigation in urban areas, the autonomous vehicle has to accomplish some objectives that have opposed effects on its behavior. For instance, the vehicle tries to progress along the road with a near-constant speed, but have to slow down when it encounters slower vehicles in front of it. Balance those objectives in such a way to produce a desired behavior is a time-consuming task, and can become very hard as more features are added to the problem.

1.1 Thesis objective and main contributions

This work focus on decision-making for autonomous vehicles in the presence of uncertainty and learning driving behavior through real data. The main approaches presented throughout this thesis to solve the decision-making problem apply variants of the well-known Partially Observable Markov Decision Process (POMDP) (KAELBLING; LITTMAN; CASSANDRA, 1998) and Inverse Reinforcement Learning (IRL) (NG; RUSSELL *et al.*, 2000) algorithms, which are properly defined in Chapter 2. POMDP a powerful tool when the true states of the system are unknown and can only be estimated through sequential observations (RUSSELL, 2010), while IRL focuses on learning the underlying reward function from observed demonstrations or expert trajectories. The main hypothesis of this thesis is that *the combination of the both approaches can be used to compute optimal actions that imitate the human behavior, which is crucial for autonomous vehicles to be socially integrated to urban roads in the presence of human drivers*.

Artificial Intelligence (AI) has been proved to be a powerful tool to address the problem of motion planning for autonomous driving, avoiding the need of manually defining complex behaviors through mathematical models. Deep Learning (DL) can be applied to this domain in order to learn a policy directly from data, in a process known as end-to-end learning. The work of Pomerleau (1989) pioneered this area. He uses an Artificial Neural Network (ANN) to produce steering commands directly from camera images and laser scans. The employed ANN has only three layers, which can be considered a tiny network for nowadays. With advances of computer processing capabilities, specially due to the development of moderns Graphics Processing Unit (GPU), ANN with greater complexity could be applied to this domain. In an earlier study, Bojarski *et al.* (2016) collect images from three cameras mounted on the vehicle in different positions. These images are used to train a Convolutional Neural Network (CNN) to output the steering wheel angle to keep the vehicle on the road. The authors highlight that such approach can avoid engineering the entire planning pipeline for autonomous driving, which can be replaced by only one CNN. Nevertheless, end-to-end approaches require a large amount of data to learn efficient policies. Moreover, it cannot provide insights into why certain decisions are being made, since the learned model is difficult to be interpreted in an human perspective. IRL often requires fewer expert demonstrations compared to training deep neural networks, which can be data-hungry. Also, IRL can capture the implicit safety and ethical considerations of human drivers.

Instead of learning a policy directly from data, some methods are dedicated to learn a part of the model, which is optimized in order to compute optimal policies. Dynamic Bayesian Networks (DBN) (RUSSELL, 2010) can be used to model relationships between variables using probabilistic dependencies. In the context of autonomous driving, they can capture uncertainties and dependencies between different aspects of the environment. Gindele, Brechtel and Dillmann (2015) apply a DBN to describe physical relationships between vehicles and other driving elements in a non-signalized intersection, and uses an Expectation Maximization (EM) approach

for learning the models integrated in the DBN. However, DBNs might not directly capture human preferences. On the other hand, IRL can learn from human demonstrations, making it more aligned with capturing the nuances of human driving.

This thesis is organized as a collection of papers, in which each chapter presents its own problem definition and contributions by covering common scenarios encountered in the autonomous driving domain:

- Chapter 3 presents a decision-making framework for autonomous vehicles at traffic light signs in the presence of uncertain measurements and speed rate constraints for comfort assurance using POMDP. The solution of the POMDP model is an optimal policy, which gives the optimal speed to be followed by the autonomous vehicle.

- Chapter 4 details a POMDP model for decision-making for autonomous driving in multi-lane roads in the presence of other vehicles. The model can estimate surrounding vehicles ongoing lane changes in order to anticipate to their maneuvers. The output of the computed policy is combination of longitudinal and lateral actions, which enable the vehicle to adapted its speed and to perform lane changes as well.

- Chapter 5 presents an approach for designing autonomous vehicles behavior using learning from demonstration. A variation of the IRL algorithm is proposed in order to deal with continuous state spaces. The experiments are performed in a merging scenario considering real data, showing that the proposed method can generate trajectories similar to the ones executed by human drivers.

- Chapter 6 extends Chapter 5 by considering uncertainty in surrounding vehicles behavior. A POMDP model is used to plan actions by estimating the intention of other vehicles of giving the right way to the autonomous vehicle to merge onto the target lane. Results show that the proposed approach compares favorably to deterministic methods, in which surrounding vehicles latent intentions are not considered in the IRL problem formulation.

Moreover, Chapter 7 highlights key aspects of this work and addresses future work as well.

1.2 Publications

During the thesis formulation, we have published and submitted papers to relevant journals and conferences in the autonomous vehicles and robotics fields, either as first authors or in collaboration with other colleagues:

Published journal articles:

"*Sparse Road Network Model for Autonomous Navigation Using Clothoids*", Júnior A. R. da Silva, Iago P. Gomes, Denis F. Wolf, Valdir Grassi Jr. Published in IEEE Transactions on Intelligent Transportation Systems, Volume: 23, Issue 2, February 2022 [Qualis A1, Impact factor: 5.293].

Accepted journal articles:

"*Autonomous driving of trucks in off-road environment*", Kenny A. Q. Caldas, Filipe M. Barbosa, Júnior A. R. da Silva, Tiago C. dos Santos, Iago P. Gomes, Luis A. Rosero, Denis F. Wolf, Valdir Grassi Jr. Accepted in Journal of Control, Automation and Electrical Systems, April 2023 [Qualis A4].

Journal articles under review:

"*Interaction-aware Decision-making on Multi-lane Roads for Autonomous Driving*", Júnior A. R. da Silva, , Valdir Grassi Jr, Denis F. Wolf. Submitted to Engineering Applications of Artificial Intelligence, May 2023.

"*Maximum Entropy Inverse Reinforcement Learning using Monte Carlo Tree Search for Autonomous Driving*", Júnior A. R. da Silva, Valdir Grassi Jr, Denis F. Wolf. Submitted to IEEE Transactions on Intelligent Transportation Systems, May 2023.

"*Learning Driving Behavior for Autonomous Vehicles in Partially Observable Environments*", Júnior A. R. da Silva, Valdir Grassi Jr, Denis F. Wolf. Submitted to IEEE Transactions on Intelligent Vehicles, May 2023.

"*Scheduling System for multiple self-driving cars using K-means and Bio-inspired optimization algorithms*", Clenio Silva, Tiago C. dos Santos, Iago P. Gomes, Júnior A. R. da Silva, Denis F. Wolf, Raulcézar Alves, Jeferson Souza. Submitted to SN Computer Science, April 2023.

Published conference papers:

"*Route Scheduling System for Multiple Self-driving Cars Using K-means and Bio-inspired Algorithms*", Clenio Silva, Tiago C. dos Santos, Iago P. Gomes, Júnior A. R. da Silva, Denis F. Wolf, Raulcézar Alves, Jeferson Souza. Publish in International Conference on Engineering Applications of Neural Networks, 2022, Creta, p. 27-39.

"*A neural path planner based on sensor fusion in the bird's eye view representation space for mapless autonomous driving*", Luis A. Rosero, Júnior A. R. da Silva, Denis F. Wolf, Fernando S. Osório. IEEE Latin American Robotics Symposium (LARS), 2022, São Bernardo do Campo, p. 181.

"*Decision Making for Autonomous Vehicles at Signalized Intersection under Uncertain Traffic Signal Phase and Timing Information*", Júnior A. R. da Silva, Valdir Grassi Jr, Denis F. Wolf. IEEE 20th International Conference on Advanced Robotics (ICAR), 2021, Ljubljana, p. 619.

"*Continuous Deep Maximum Entropy Inverse Reinforcement Learning using online POMDP*", Júnior A. R. da Silva, Valdir Grassi Jr, Denis F. Wolf. IEEE 20th International Conference on Advanced Robotics (ICAR), 2019, Belo Horizonte, p. 382.

"*Advanced Driver Assistance System Based on Automated Routines for the Benefit of Human Faults Correction in Robotics Vehicles*", Diego Bruno, Tiago C. dos Santos, Júnior A. R. da Silva, Denis F. Wolf, Fernando S. Osório. IEEE Latin American Robotics Symposium (LARS), 2018, João Pessoa, p. 112.

Preprint articles:

"*A software architecture for autonomous vehicles: Team LRM-B entry in the first carla autonomous driving challenge*", Luis A. Rosero, Iago P. Gomes, Júnior A. R. da Silva, Tiago C. dos Santos, Angelica T. M. Nakamura, Jean Amaro, Denis F. Wolf, Fernando S. Osório. arXiv preprint arXiv:2010.12598, 2020.

1.3 Honors and Awards

During the period covered by this thesis, the author has participated in additional activities. An important activity was the participation in the *CARLA Autonomous Driving Challenge*[1], an international challenge for autonomous vehicles in simulated urban environments. The team LRM, of which the researcher was part, won first place in three of the four categories available in the challenge. The team won second place in the only category in which it did not win:

"*First Place Achievement of Track 1*", CARLA Autonomous Driving Challenge 2019 - July, 2019.

"*First Place Achievement of Track 3*", CARLA Autonomous Driving Challenge 2019 - July, 2019.

"*First Place Achievement of Track 4*", CARLA Autonomous Driving Challenge 2019 - July, 2019.

"*Second Place Achievement of Track 2*", CARLA Autonomous Driving Challenge 2019 - July, 2019.

The fact that a Brazilian team was the main winner of the competition got relevant media coverage, as can be seen here, here, here and here.

[1] https://carlachallenge.org/

CHAPTER

2

BACKGROUND

The theoretical background of Markov Decision Process (MDP) and Partially Observed MDP is given in this chapter. Their mathematical framework comes along with examples in order to provide a better understanding of their applications. Also, the basis of Maximum Entropy Inverse Reinforcement Learning, one of most popular algorithms used in IRL, is presented.

2.1 Markov Decision Process

During navigation, the ego vehicle must adopt some tactical behaviors in order to safely reach its destination, such as avoiding static and dynamic obstacles. Considering only static obstacles, the ego vehicle needs to plan a trajectory to avoid them, while guaranteeing some level of comfort for the passengers. This task becomes more complex when other traffic participants are considered. For example, intersection negotiation and overtaking require a higher level of reasoning than deviates from a static obstacle on the road. To achieve such level of reasoning, the ego vehicle must consider its future actions as well as other vehicles' motion. This prediction comes along with uncertainty, assuming that communication between vehicles is not available. In order to deal with the inherent uncertainty, the problem can be modeled as a Markov Decision Process (MDP) (BRECHTEL; GINDELE; DILLMANN, 2011).

MDP is a mathematical framework to solve problems subject to uncertainty (RUSSELL, 2010). It is defined by the tuple $(\mathcal{S},\ \mathcal{A},\ T,\ R,\ \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ are the state and action space, respectively. When taking action $a \in \mathcal{A}$ in state $s \in \mathcal{S}$, the agent reaches the state $s' \in \mathcal{S}$. The conditional probability function $T(s,\ a,\ s') = \Pr(s'|s,\ a)$, which describes the probability of reaching s' from s after taking a, models the uncertainty related to state transitions. $R(a,\ s)$ is the expected reward when taking action a in s. MDP aims to compute a policy $\pi^* : s \rightarrow a$ that maximizes the expected sum of discounted rewards.

Computing π^* is not straightforward since the agent is in a stochastic environment.

Therefore, not only $R(s,\ a)$ has influence on π^*, but also $T(s,\ a,\ s')$. In other words, π^* gives the best action to take for every $s \in \mathcal{S}$, which also depends on the dynamics of the system. In this way, π^* attempts to drive the agent to states that are more *valuable* than others. The *value* or *utility* of a state is given by the expected sum of discounted rewards in time

$$V^{\pi}(s) = \mathrm{E}\sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(s_t)), \tag{2.1}$$

where s_t are the reachable next states according to π for $t > 0$, and $\gamma \in [0,1)$ is the discount factor, which prioritizes immediate rewards. The optimal policy is the one that gives the maximum utility in s,

$$\pi^*(s) = \underset{\pi}{\operatorname{argmax}}\ V^{\pi}(s) \tag{2.2}$$

where the optimal utility of s is given by

$$V^*(s) = \max_{a \in \mathcal{A}} Q^*(s,a). \tag{2.3}$$

$Q^*(s,a) = \sum_{s'} T(s,\ a,\ s')[R(s,\ a) + \gamma V(s')]$ is the Q-value, which is the optimal utility for s when executing action a. Therefore,

$$V^*(s) = \max_{a \in \mathcal{A}} \sum_{s'} T(s,\ a,\ s')[R(s,\ a) + \gamma V(s')] \tag{2.4}$$

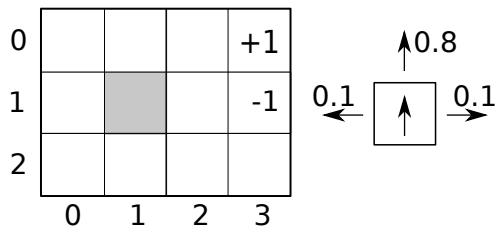
which is the well-known *Bellman equation* (BELLMAN, 1957).

As it can be noted, the Bellman equation computes the optimal policy of only one state s. Thus, n equations are required to solve a problem with n states. Since the system is composed of nonlinear equations (due to the "max" operand), the problem can be solved using an iterative approach. First, the utility of all states are initialized with arbitrary values. After that, $V(s)$ is updated according to possible actions $a \in \mathcal{A}$ and all the reachable states s' from s:

$$V_{i+1}(s) = \max_{a \in \mathcal{A}} \sum_{s'} T(s,\ a,\ s')[R(s,\ a) + \gamma V_i(s')] \tag{2.5}$$

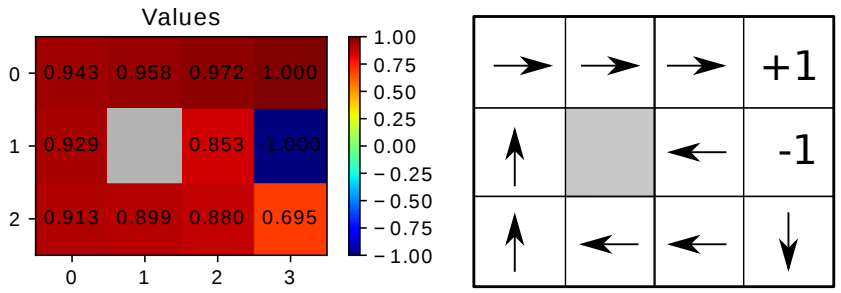
Each Bellman equation is updated until the stop criteria $|V_{i+1}(s) - V_i(s)| < \varepsilon$ is reached for each s, where ε is the threshold value. In this process, γ has an convergence effect, since an

Figure 2 – 3×4 grid world.



Source: Elaborated by the author.

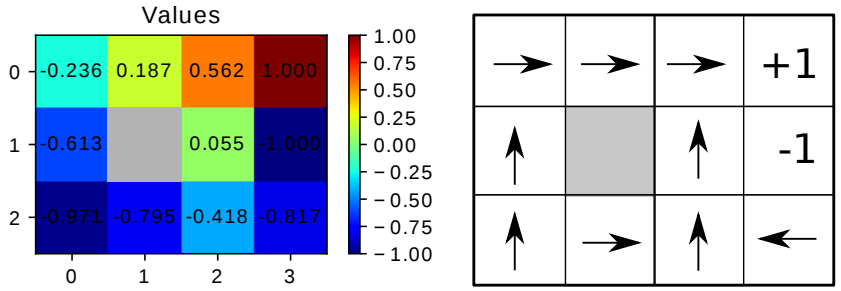
Figure 3 – $R_{\text{int}} = -0.01$ for intermediate states.



(a) Values. (b) Policy.

Source: Elaborated by the author.

Figure 4 – $R_{\text{int}} = -0.3$ for intermediate states.



(a) Values. (b) Policy.

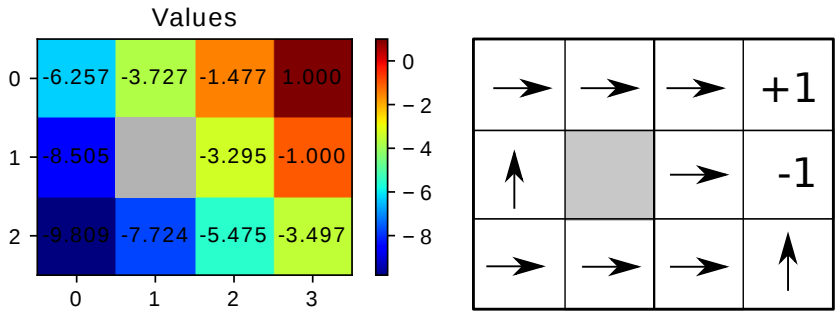
Source: Elaborated by the author.

infinite horizon solution is sought. This is known as the *value iteration* algorithm, one of the most popular algorithms applied to MDP solving (RUSSELL, 2010).

To illustrate MDP, the following example is considered: a robot lives in a 3×4 grid world (Figure 2), in which $\mathcal{A} = \{Up, Down, Left, Right\}$ are the possible movements it can execute. However, the transition between states is uncertain: the robot can reach the intended state with probability 0.8, but with probability 0.2 the robot moves at right angles to the chosen action. When it collided with a wall it remains on the current cell. The two terminal states have rewards +1 and -1, i. e. the robot leave the grid world when it reaches one of the terminal states.

Figures 3, 4 and 5 show the effect of changing the rewards R_{int} of intermediate states. Figure 3 depicts the values and the policy when $R_{\text{int}} = -0.01$. Since R_{int} is not too much negative, the policy is too conservative, avoiding the negative terminal state. If $R_{\text{int}} = -0.3$, staying in the grid world becomes more "painful", so the policy is less conservative in order to reach the positive terminal state as soon as possible (Figure 4). When $R_{\text{int}} = -1.8$, it becomes too much "painful" staying in the grid world and, consequently, the policy attempts to find a terminal state, no matter what it is (Figure 5). Thereby, the agent's behavior can be changed by adjusting $R(s, a)$.

Figure 5 – $R_{\text{int}} = -1.8$ for intermediate states.



(a) Values. (b) Policy.

Source: Elaborated by the author.

2.2 Partially Observable Markov Decision Process

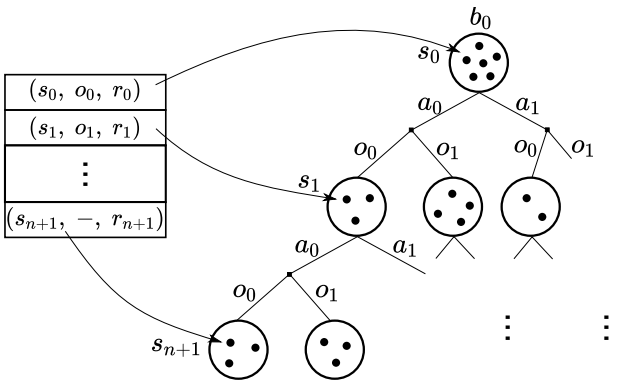
In the previous section, it is assumed that the agent can completely observe its current state. In a Partially Observable Markov Decision Process (POMDP), the state space are not fully observable. Instead, the state s is only partially observed given an observation $o \in \mathcal{O}$, where $\mathcal{O}$ is the agent's observation space. After taking action a, the agent make an observation in s'. The observation in s' is given by a conditional probability function $\mathcal{Z}(s', a, o) = \Pr(o|s', a)$. Unlike MDPs, it is not possible to compute a policy that maps states into actions because the agent does not know the state in which it is. Thus, the optimal policy $\pi^* : b \rightarrow a$ maps a *belief* $b \in \mathcal{B}$ into an action $a \in \mathcal{A}$, where b is a probability distribution over $\mathcal{S}$, and $\mathcal{B}$ is the belief space. Using value iteration algorithm does not work for POMDPs, since $\mathcal{B}$ is continuous even for discrete state spaces. Like states in MDP, each belief b has its value or utility $V(b)$, which can be approximated by the linear, piecewise function:

$$V^*(b) = \max_{\alpha \in \Gamma} \{\alpha.b\}, \tag{2.6}$$

where Γ is the finite set of vectors α called α-vectors (KAELBLING; LITTMAN; CASSANDRA, 1998). Each element of α is the utility of executing a fixed conditional plan starting from s. The optimal policy corresponds to the best α-vector in the current belief b according to (2.6). However, evaluating the conditional plans becomes more costly as the planning depth increases. The exact optimal policy computation becomes intractable even for problems with dozes of states (RUSSELL, 2010). Therefore, methods that are able to compute approximate solutions have been developed in literature (LITTMAN; CASSANDRA; KAELBLING, 1995; KURNIAWATI; HSU; LEE, 2008; SILVER; VENESS, 2010; SOMANI *et al.*, 2013; KURNIAWATI; YADAV, 2016). In this work, the online solver Partially Observable Monte Carlo Planning (POMCP) (SILVER; VENESS, 2010) is used to compute policies in POMDPs.

POMCP maintains a belief tree $\mathcal{T}$ by sampling *episodes* h, which are composed of a sequence of quadruples (s, a, o, r)(Figure 6). The models T, $\mathcal{O}$ and R must not be given explicitly, instead POMCP samples an action a and an initial state s from the initial belief b_0. A

Figure 6 – Partially Observable Monte Carlo Planning (POMCP). Adapted from (KURNIAWATI; YADAV, 2016).



Source: Elaborated by the author.

generative model is responsible for generating an observation o, a reward r and a next state s'. Since the policy is solved by sampling episodes, POMCP can handle either discrete or continuous states. Each node in $\mathcal{T}$ is a belief b that can be reached from the initial belief b_0. In this way, the value equation is defined as

$$V^*(b) = \underset{a \in \mathcal{A}(E,b)}{\operatorname{argmax}} \hat{Q}(b,\ a), \tag{2.7}$$

where E is the set of edges in $\mathcal{T}$, and $\mathcal{A}(E,b)$ is the set of actions that have been used to expand b. The value $\hat{Q}(b,\ a)$ is the estimated Q-value, given by

$$\hat{Q}(b,\ a) = \frac{1}{|H_{(b,\ a)}|} \sum_{h \in H_{(b,\ a)}} V(h,\ l). \tag{2.8}$$

$H_{(b,\ a)}$ is the set of all sampled episodes associated with all paths in $\mathcal{T}$ that contains $(b,\ a)$, $|.|$ is the size of a set, l is depth level of b in $\mathcal{T}$, and $V(h,\ l)$ is the value of an episode h started from the l^{th} element, given by

$$V(h,\ l) = \sum_{i=l}^{|h|} \gamma^{i-l} R(h_i.s,\ h_i.a). \tag{2.9}$$

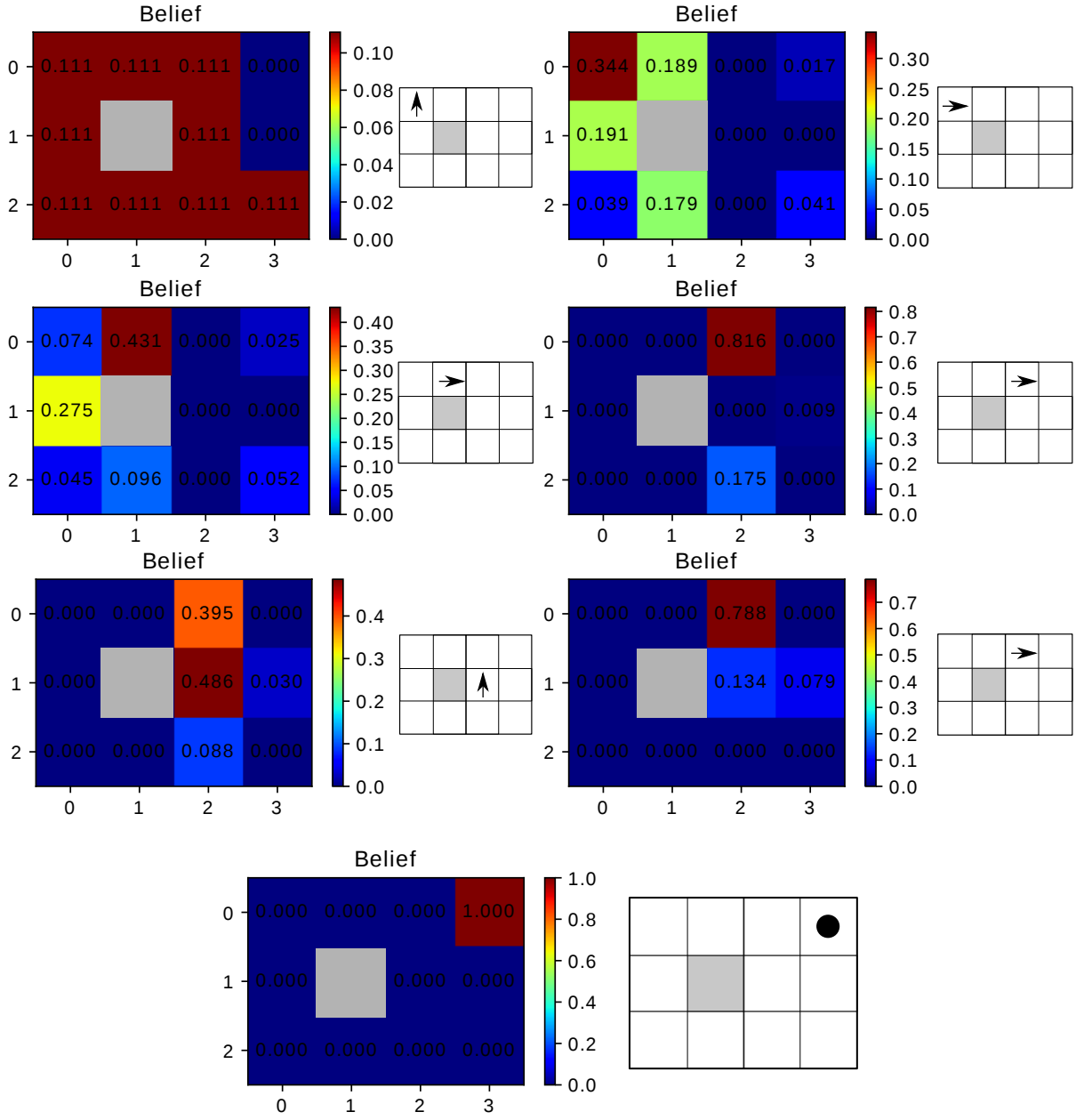
Now, it remains to defined how to select an action to be executed at b. First, POMCP executes all available action until all $\mathcal{A}(E,b)$ have been covered. After that, POMCP selects an action according to

$$a = \underset{a \in \mathcal{A}}{\operatorname{argmax}} \left(\hat{Q}(b,\ a) + c \sqrt{\frac{\log(|H_b|)}{|H_{(b,\ a)}|}} \right) \tag{2.10}$$

where H_b is the set of episodes associated with b, and c is a scalar value that determines the ratio between exploration and expectation.

The next example illustrates how POMCP works in practice. Now, in the grid world of Section 2.1, the agent no longer can observe its current state: it can only observe the number

Figure 7 – Online POMDP solving using POMCP. As more observations are gathered, the agent becomes more confident about its true current state.



Source: Elaborated by the author.

of walls of a cell. In the grid world, the cells (0, 0), (0, 1), (0, 3), (1, 0), (2, 0), (2, 1) and (2, 3) have 2 walls, whereas the other ones have 1 wall. Figure 7 depicts the steps performed by the agent from cell (0, 0) until it reaches the positive terminal state. Despite being in cell (0, 0) at the beginning, the agent beliefs it is in any of the non-terminal states. The number inside the cells correspond to the probability of being in that cell. The arrow indicates which action is computed by POMCP. After sampling episodes for 1 second, the agent perform an action, with $c = 200$ to prioritizes exploration. As it can be seen, the agent reaches the positive terminal state with few

steps.

Table 1 presents the average number of steps, how many times the agent succeeds in solving the policy and how many times it fails. Ten trajectories are performed for each initial state. One key thing to note is that a policy is computed for each belief. As shown in Figure 7, the optimal policy for the initial belief is $a = Up$. However, when the agent is in cell (2, 3) it fails in the most of the times because going up tends to lead it to the negative terminal state. Table 2 illustrate the case when the agent receive noisy observations of the environment: it observe the right number of walls with probability 0.9. Although the number of times the agent fails increases, it still can satisfactorily perform good policies, even with non-perfect observations.

Table 1 – POMDP solving considering perfect observation.

	Avg steps	Win	Lose
(0,0)	9.0	10	0
(0,1)	8.9	10	0
(0,2)	3.6	9	1
(1,0)	12.1	10	0
(1,2)	3.7	8	2
(2,0)	18.4	8	2
(2,1)	14.1	9	2
(2,2)	4.3	7	3
(2,3)	5.5	2	8

Source: Elaborated by the author.

Table 2 – POMDP solving considering noisy observation.

	Avg steps	Win	Lose
(0,0)	12.2	10	0
(0,1)	8.9	10	0
(0,2)	4.2	9	1
(1,0)	11.5	9	1
(1,2)	5.0	9	1
(2,0)	10.5	7	3
(2,1)	15.2	7	3
(2,2)	9.8	4	6
(2,3)	4.9	4	6

Source: Elaborated by the author.

2.3 Inverse Reinforcement Learning

The ego vehicle must act according to some traffic rules and social acceptance. This behavior might be passed to the ego vehicle by tuning a reward function. However, this process

can become a tedious, error prone task, since the sought behavior is governed by many rules. To overcome this issue, the reward function can be learned from expert trajectory demonstrations via IRL.

A trajectory $\zeta = ((s_1, a_1), (s_2, a_2), ...)$ is a set of state-action pair (s,a). The total reward of a trajectory is a combination of features $\mathbf{f}_\zeta = \sum_{s\in\zeta} \mathbf{f}_s$ and weight parameters θ, which maps the features into rewards:

$$R(\zeta) = \sum_{\zeta} R_\theta(\mathbf{f}_\zeta). \tag{2.11}$$

Nevertheless, ambiguity may occur when determining θ. For example, a policy may be explained by different reward functions and vice-versa. Moreover, the trajectories performed by the expert are expected to be optimal, which imposes an important constraint to the problem. One solution is assuming expert trajectories as noisy optimal trajectories sampled from a distribution

$$p(\zeta) \propto e^{R(\zeta)}, \tag{2.12}$$

which gives that the likelihood of a expert trajectory is higher for higher rewards.

Ziebart *et al.* (2008) applied this principle to develop the well-known *Maximum Entropy Inverse Reinforcement Learning*, in which the reward model is learned from the maximization of the likelihood of the observed data under maximum entropy,

$$\begin{aligned}
\theta^* &= \text{argmax}_\theta L(\theta) \\
&= \text{argmax}_\theta \frac{1}{M} \log \Pr(\{\zeta\}|\theta) \\
&= \text{argmax}_\theta \frac{1}{M} \log \prod_{\zeta} \Pr(\zeta|\theta) \\
&= \text{argmax}_\theta \frac{1}{M} \sum_{\zeta} \log \Pr(\zeta|\theta) \\
&= \text{argmax}_\theta \frac{1}{M} \sum_{\zeta} \log \frac{e^{R_\theta(\zeta)}}{Z} \\
&= \text{argmax}_\theta \frac{1}{M} \sum_{\zeta\in\zeta_{\text{demo}}} R_\theta(\zeta) - \log Z,
\end{aligned} \tag{2.13}$$

in which M is the number of expert trajectories and

$$Z = \sum_{\zeta\in\zeta_{\text{back}}} e^{R_\theta(\zeta)} \tag{2.14}$$

is the partition function, where ζ_{back} are trajectories sampled from the background distribution given θ. The gradient $\nabla_\theta L$ can be computed as

$$\nabla_\theta L = \frac{1}{M} \sum_{\in\zeta_{\text{demo}}} \mathbf{f}_\zeta - \frac{1}{Z} \sum_{\zeta\in\zeta_{\text{back}}} \left(e^{R_\theta(\zeta)} \frac{dR_\theta(\zeta)}{d\theta} \right) \tag{2.15}$$

When $R_\theta(\mathbf{f}_\zeta)$ is linear with respect to θ, the partition function can be exactly computed by using a dynamic programming algorithm, avoiding the need to sample many possible trajectories to estimate Z:

$$\begin{aligned}\nabla_\theta L &= \frac{1}{M}\sum_{\zeta\in\zeta_{\text{demo}}}\mathbf{f}_\zeta - \sum_{\zeta\in\zeta_{\text{back}}}\frac{e^{R_\theta(\zeta)}}{Z}\mathbf{f}_\zeta \\ &= \frac{1}{M}\sum_{\zeta\in\zeta_{\text{demo}}}\mathbf{f}_\zeta - \sum_{\zeta\in\zeta_{\text{back}}}\Pr(\zeta|\theta)\mathbf{f}_\zeta.\end{aligned} \tag{2.16}$$

Since trajectories ζ are a sequence of states,

$$\nabla_\theta L = \frac{1}{M}\sum_s \mathbf{f}_s - \sum_s \Pr(s|\theta)\mathbf{f}_s, \tag{2.17}$$

where $\Pr(s|\theta)$ is the state visitation frequency of state s and can be determined by dynamic programming:

$$\mu_{t+1}(s) = \sum_a\sum_{s'}\mu_t(s')\pi(a,\, s')T(s,\, a,\, s') \tag{2.18}$$

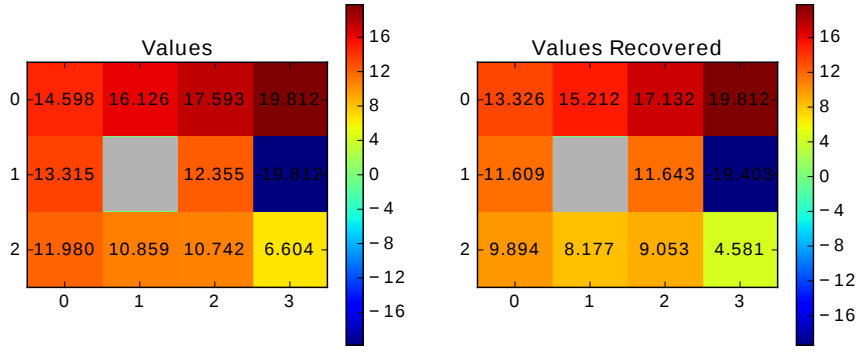
where μ_t denotes the probability of visiting s at t which gives that

$$\Pr(s|\theta) = \sum_t \mu_t(s). \tag{2.19}$$

It is important to note that an optimal policy must be found at each iteration in order to compute the state visitation frequency. With the gradient, θ can be update at each iteration using gradient descent optimization.

The MDP presented in Section 2.1 with $R_{\text{int}} = -0.04$ is used to illustrate the Maximum Entropy IRL algorithm. But now, the agent is trying to recover the reward model from expert demonstrations. The initial state of the demonstrations are sampled uniformly from the non-terminal states. Since the algorithm considers trajectories with same length, the model must be slightly changed: now, when the agent reaches a terminal state, it remains there until a predefined

Figure 8 – Maximum Entropy IRL applied to the 3×4 grid world.



Source: Elaborated by the author.

number of steps is concluded. As depicted in Figure 8, the values are satisfactorily recovered using 400 expert demonstrations with length 10.

A drawback of Maximum Entropy IRL algorithm is that it assumes the reward function as a linear combination of weight parameters and features. Therefore, it can fails to learn complex non-linear functions. Moreover, an MDP must be solved at each iteration in order to compute the gradients, which can become very costly or even intractable for problems with large state space.

2.4 Final Considerations

This chapter shows how MDP can be used to solve problems in which uncertainty can not be neglected. Moreover, it shows how the optimal behavior of the agent can be changed by adjusting the reward model. Considering a partially observable environment, the optimal policy is estimated using the online POMDP solver POMCP. Also, a reward function is properly recovered using Maximum Entropy IRL. Next chapters present how POMDP and IRL can be applied to the autonomous driving problem by dealing with uncertainty as well as by mimicking human drivers behavior.

CHAPTER

3

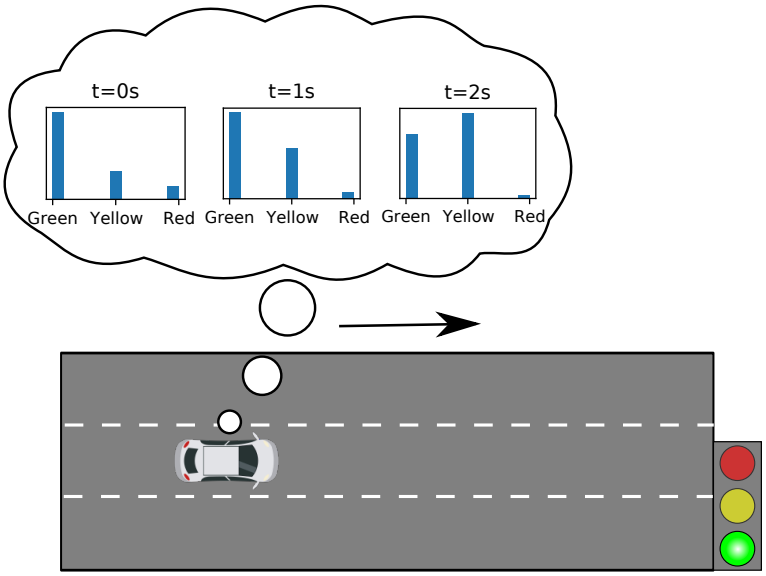
DECISION MAKING FOR AUTONOMOUS VEHICLES AT SIGNALIZED INTERSECTION UNDER UNCERTAIN TRAFFIC SIGNAL PHASE AND TIMING INFORMATION

In real environments, autonomous vehicles must be able to deal with uncertainties related to the measurements provided by their perception system. Not taking such perception uncertainties into account can lead the vehicle to take erroneous decisions and cause accidents. Excessive speed rate change variations and red light crossing at signalized intersections are special cases of this problem. This chapter presents a decision making for autonomous vehicles that considers uncertainty in timing information, but also in traffic light color (signal phase) measurement at signalized intersections. A Partially Observable Markov Decision Problem (POMDP) model is proposed in order to deal with the problem of partially observability of both signal phase and timing. By incorporating the perception system uncertainty, the POMDP model can reliably predict signal phase transitions, avoiding too reactive behaviors and running red lights as well. Results show that the proposed POMDP model is able to estimate the true values of the signal phase and timing as more observations are gathered, which allows better decisions when compared to deterministic approaches.

3.1 Introduction

During navigation in urban areas, self-driving cars, as well as human drivers, must deal with traffic lights placed at signalized intersections. The purpose of such traffic signs is the organization of the traffic flow. Since traffic lights have a predefined transition cycle (red-green-yellow-red), vehicles must be aware of such transitions, so they do not adopt excessive reactive behaviors, mainly during yellow signals (HORST *et al.*, 1988).

Figure 9 – An autonomous vehicle is approaching a signalized intersection. It must be reason about future transitions in the traffic light phase in order to plan its velocity. This task becomes more difficult due to noisy sensor measurements and unknown signal timing.



Source: Elaborated by the author.

In the specific automated driving domain, the traffic signal can be perceived either by sensors attached to the vehicle or through direct communication. The latter imposes the autonomous vehicle to have access to a vehicle-to-infrastructure communication (V2I), which may be a very optimistic assumption (KATRAKAZAS *et al.*, 2015). In this sense, the use of sensors attached to the vehicle, such as monocular and stereo vision cameras, becomes a cheaper and viable option when dealing with the problem of traffic signal detection (LEVINSON *et al.*, 2011; BEHRENDT; NOVAK; BOTROS, 2017). Nevertheless, sensors may provide noisy measurements, which may be caused by color tone shifting and halo disturbances, occlusion and partial occlusion, incomplete shapes because of malfunctioning or dirty lights, and false positives (JENSEN *et al.*, 2016). This can lead the vehicle to take erroneous decisions, such as performing trajectories with excessive speed rate change variations in response to noisy measurements and, in the worst case, running red lights and causing accidents. Therefore, a robust automated vehicle system must take into account uncertainties and passengers' comfort during the motion planning phase (Fig. 9).

Most works in the literature addresses the problem of motion planning at signalized intersection from the fuel consumption perspective, also known as *eco-driving*. Asadi and Vahidi (2010) propose a predictive cruise control that takes into account signal phase and timing to improve fuel usage and to reduce trip time. The problem is solved using Model Predictive Control (MPC) (CAMACHO; ALBA, 2013), in which the constraints and objectives of the problem can be modeled in an intuitive way. By considering multiple signalized intersections and vehicle motion during turning, Huang and Peng (2017) propose a sequential convex optimization

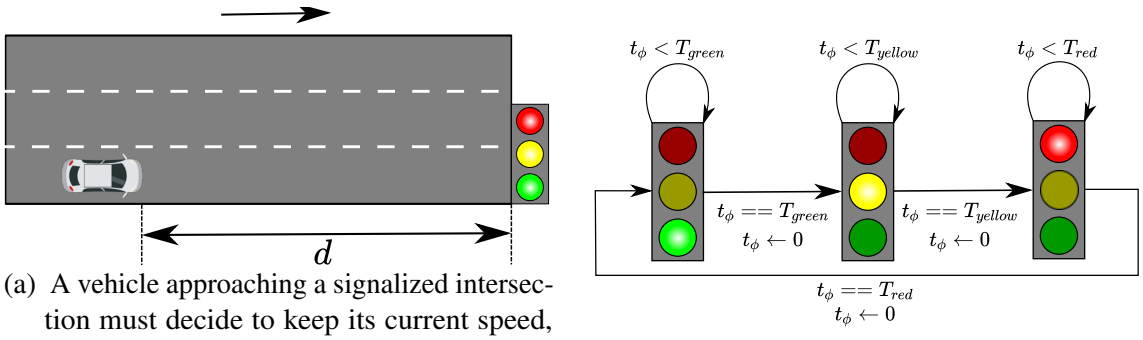
formulation. It allows the solver to find a local optimal solution without suffering from the curse of dimensionality, since the main problem is divided into sub-problems. Despite being able to provide efficient solutions, none of those works consider uncertainties in Signal Phasing and Timing (SPaT) estimation.

A particular traffic light SPaT can vary during different periods of the day to adapt to traffic conditions (MAHLER; VAHIDI, 2012). Therefore, considering deterministic SPaT can become a too simplistic approach when dealing with such problem. This issue is addressed by Mahler and Vahadi (MAHLER; VAHIDI, 2012; MAHLER; VAHIDI, 2014) by assuming that SPaT is uncertain even for fixed-time signals. They estimate the traffic signal phase by using conditional probability functions and by considering that the average duration of red and green signals is known. Another possible approach is to incorporate the uncertainty directly into the model, without previously estimating it. A Reinforcement Learning (RL)-based control (SUTTON; BARTO, 2018) for connected vehicles is presented by Zhou, Yu and Qu (2019). Although SPaT of traffic lights are randomly set at each episode, the agent can fully observe the traffic light color and light cycle. Sun, Shen and Moura (2018) propose a robust optimal eco-driving control by defining the red-light duration as a random variable and the optimal solution is computed via Dynamic Programming (DP) (BELLMAN, 2010). In a posterior work (SUN *et al.*, 2020), the authors extend the application to connected vehicles, in which the optimal control is applied to a queue of vehicles approaching the intersection. By estimating the red-light duration, the vehicle can chose whether to stop or not: if the red-light cycle is finishing as it approaches the intersection, the vehicle can cross the intersection without stopping. Nevertheless, not considering uncertainty in the light observation can lead the vehicles to take erroneous actions. Also, those works assume a long acceleration range (-4 m/s^2 to 2 m/s^2) and no minimal distance to perceive the traffic light.

The contribution of this chapter is a decision-making framework for autonomous vehicles at traffic light signs in the presence of uncertain measurements and speed rate constraints for comfort assurance. In order to deal with imperfect sensor information, the problem is modeled as a Partially Observable Markov Decision Process (POMDP), which is a powerful tool when the true states of the system are unknown and can only be estimated through sequential observations (RUSSELL, 2010). The signal phase as well as its timing are considered as partial observable states, which makes the vehicle aware of possible signal transitions, specially during the end of a green signal phase, enhancing robustness when receiving information from noisy sensors. Moreover, the absolute speed rate range is very limited in order to assure comfort to passengers. The solution of the POMDP model is an optimal policy, which gives the optimal speed to be followed by the autonomous vehicle with respect to its current belief about SPaT.

The remainder of this chapter is organized as follows: the problem covered in this work is stated in Section 3.2; Section 3.3 describes the POMDP model and how it can be approximately solved in real-time using particle sampling; the results are presented and discussed in Section

Figure 10 – Problem Statement.



(a) A vehicle approaching a signalized intersection must decide to keep its current speed, to slow down or to accelerate.

(b) Traffic light color transitions.

Source: Elaborated by the author.

3.4; finally, remarks and future work are presented in Section 3.5.

3.2 Problem Statement

A controlled vehicle approaches an intersection signalized by a traffic light, which has three different states: *GREEN*, *YELLOW* and *RED*, as shown in Fig. 10a, where d is the distance to the intersection. The states' transitions are depicted in Fig. 10b, where t_ϕ is the elapsed time since the last state transition, and T_{green}, T_{yellow} and T_{red} are the expected duration times for *GREEN*, *YELLOW* and *RED* staying activated, respectively. From a high-definition map, the vehicle have access to d, but can only observe the light color from a minimum distance d_{min}. When the sign is red, the vehicle is not able to run through the intersection and must wait until the sign has turned green. Thus, the vehicle's goal is to keep its reference speed during green lights and not to cross the intersection during red lights. In the presence of perfect sensor measurements or V2I communication, the problem can be solved by deterministic approaches, such as the ones discussed in Section 3.1. Nevertheless, with only noisy information, the vehicle cannot determine the current state of the traffic light. Instead, it has a belief about the true state, which changes as it gathers subsequent measurements. The planned speed is given by the solution of the the POMDP optimization.

3.3 POMDP Model

This section presents the POMDP model used to compute near-optimal actions for the controlled vehicle when it is approaching a signalized intersection. Those actions are computed using the online solver POMCP (SILVER; VENESS, 2010), as detailed in Chapter 2.

3.3.1 State Space

The state space $\mathcal{S}$ includes the states of the autonomous vehicle and the traffic light in front,

$$\mathcal{S} = [d \quad v \quad \phi \quad t_\phi]^{\mathrm{T}}, \tag{3.1}$$

where d is the distance between the vehicle and the traffic light; v is the velocity of the vehicle; $\phi \in \{GREEN, YELLOW, RED\}$ is the state of the traffic light; and t_ϕ is the elapsed time since the last state transition of ϕ. While d and v are directly observable, ϕ and t_ϕ are only partially observable, which models the uncertainty in SPaT.

3.3.2 Action Space and Transition Model

The transition on the state variables related to the vehicle's longitudinal motion is constrained to the following motion model:

$$\begin{bmatrix} d' \\ v' \end{bmatrix} = \begin{bmatrix} d \\ v \end{bmatrix} + \begin{bmatrix} -v \\ a_{\mathrm{lon}} \end{bmatrix} \Delta t - \frac{1}{2} \begin{bmatrix} a_{\mathrm{lon}} \\ 0 \end{bmatrix} \Delta t^2, \tag{3.2}$$

in which a_{lon} is the speed rate change of the vehicle, Δt is the time step and d' and v' are the updated values of d and v. Low values of a_{lon} guarantee the comfort level during the travel, since the vehicle smoothly accelerates and slows down.

The traffic light state ϕ is assumed to change according to t_ϕ, as described in Fig. 10b, where

$$t'_\phi = t_\phi + \Delta t, \tag{3.3}$$

where t'_ϕ is the updated value of t_ϕ. However, t'_ϕ does not increase indefinitely: it must be reset whenever one of the following conditions are met:

$$\begin{aligned} t'_\phi &\geq \left(T_{green} + \mathcal{N}(0,\ \sigma^2)\right) \text{ and } \phi = GREEN, \\ t'_\phi &\geq \left(T_{yellow} + \mathcal{N}(0,\ \sigma^2)\right) \text{ and } \phi = YELLOW, \\ t'_\phi &\geq \left(T_{red} + \mathcal{N}(0,\ \sigma^2)\right) \text{ and } \phi = RED, \end{aligned} \tag{3.4}$$

meaning that the duration of the phase has been reached. Instead of simply assuming $t'_\phi = 0$, its value is sampled from the uniform distribution $\mathcal{U}(0, \Delta t)$. Sampling t_ϕ from $\mathcal{U}(0, \Delta t)$ adds robustness to the model, since observations are gathered at a discrete time step Δt. Also, the transition model considers that a simulated Gausian noise $\mathcal{N}(0,\ \sigma^2)$ is added to T_{green}, T_{yellow} and T_{red}, which makes the model robust to the uncertainty in the remaining time until the next state transition.

3.3.3 Observation Space and Observation Model

The observation space $\mathcal{O}$ is defined as

$$\mathcal{O} = [d_{\mathrm{obs}} \quad v_{\mathrm{obs}} \quad \phi_{\mathrm{obs}}]^{\mathrm{T}}, \tag{3.5}$$

where d_{obs}, v_{obs} and ϕ_{obs} are the observed distance to the intersection, vehicle's speed and the traffic light state, respectively. While d_{obs} and v_{obs} are fully observable states, ϕ_{obs} is a noisy measurement of the true traffic light state. In order to model the uncertainty, ϕ_{obs} is given by

$$\begin{aligned} \phi_{obs} &= \phi', \quad \text{with probability } p, \\ \phi_{obs} &\neq \phi', \quad \text{with probability } 1-p, \end{aligned} \tag{3.6}$$

where p describes the accuracy of the perception system.

3.3.4 Reward Model

The reward model gives the desired behavior of the autonomous vehicle, which must accomplish two main objectives: 1) not to run red lights and 2) maintain its reference speed when $\phi = GREEN$. Also, when $\phi = RED$, the vehicle must stop within a certain distance range d_{stop} in order to be able to keep the traffic light in its field of view, according to Fig. 11. Therefore, the reward function is given by

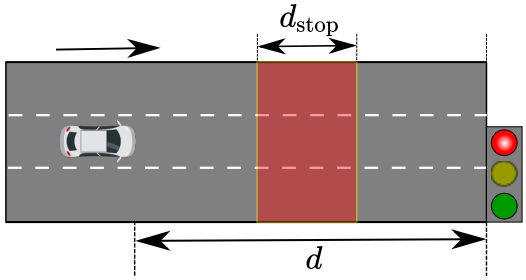
$$R(s,\, a) = R_v + R_{stop} + R_\phi + R_a, \tag{3.7}$$

where $R_v = (v - v_{ref})^2/(V_{travel})^2 + 1$ is the reward of following the reference speed v_{ref}. When d is within d_{stop} and $\phi = RED$, the vehicle must stop, which results in $v_{ref} = 0$. If this condition is not met, $v_{ref} = V_{travel}$, which is the traveling speed given by the mission planning. In order to avoid unnecessary stops, a punishment $R_{stop} = -0.9$ is given whenever the vehicle is stopped. $R_\phi = -4$ is the punishment for crossing red lights, and $R_a = \{-0.1 | (a_t - a_{t-1} \neq 0)\}$ improves comfort by penalizing changes in the chosen action.

3.4 Experimental Results

To validate the proposed approach, the experiments are conducted on the realistic simulator Car Learning to Act (CARLA) (DOSOVITSKIY *et al.*, 2017), version 0.9.10.1, which is running in synchronous mode. This means that the simulation runs at a fixed time step, which

Figure 11 – Stop zone for red lights.



Source: Elaborated by the author.

Figure 12 – Simulation environment in CARLA.

(a) Back view. (b) Bird eyes view.

Source: Elaborated by the author.

Table 3 – POMCP and POMDP model parameters.

Δt	c	γ	T_{green}	T_{yellow}	T_{red}	σ^2
1.0 s	50	0.99	10 s	4 s	20 s	2

enables more control over the experiments. In this mode, the simulation runs at 10 Hz. The system containing the simulation environment and algorithms runs on a Intel Core i7-7700 CPU with 3.60 Hz, a 15.5 GB memory RAM and a NVIDIA GeForce GTX 1050 Ti GPU. Both POMDP model and POMCP are implemented in *Python*, which provides a simpler integration to CARLA. The simulation scenario can be seen in Fig. 12.

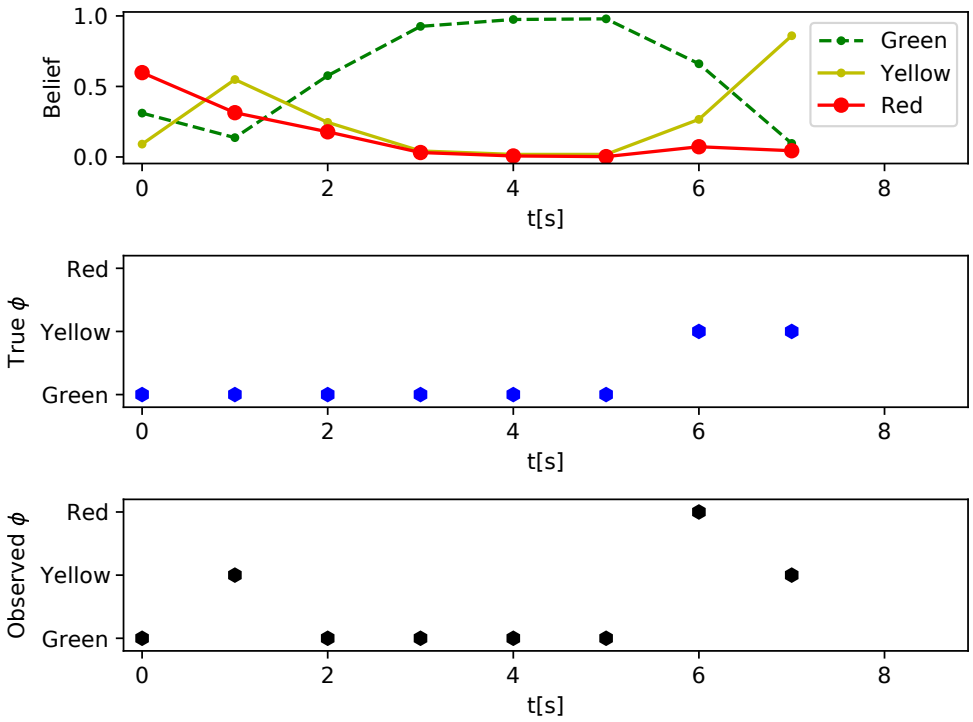
The experiments consider that the perception system can provide traffic light measurements at a $d_{\text{min}} = 50$ m distance. For simplification purposes, a real perception system that uses cameras and other sensors is not considered in this work. Instead, the ego-vehicle receives a noisy measurement according to p directly sent by CARLA. This attempts to simulate a real perception system with known accuracy. After $d <= d_{\text{min}}$, POMCP starts to run in order to output the speed to be followed by the autonomous vehicle. The parameters values used in the POMDP model and POMCP are detailed in Tab 3. Considering the values of T_{green}, T_{yellow} and T_{red}, the initial belief for ϕ is given by $p_{\text{green}} = \frac{10}{34}$, $p_{\text{yellow}} = \frac{4}{34}$ and $p_{\text{red}} = \frac{20}{34}$. The initial belief of t_ϕ is simply the uniform distribution $\mathcal{U}(0, T_\phi)$. Also, d_{stop} is considered in the range $d = [6, 12]$ m.

3.4.1 State Estimation

One of the main advantages of using POMDP is its capability of knowledge gain by gathering successive observations. In fact, the model behaves like a filter, making the particles that are less likely to be rejected. Fig. 13 and 14 depicts an example of the evolution of the beliefs about ϕ and t_ϕ, respectively.

As it can be seen in Fig. 13, some erroneous measurements are gathered by the sensors. At

Figure 13 – Signal phase estimation in the presence of noisy measurements of ϕ.



Source: Elaborated by the author.

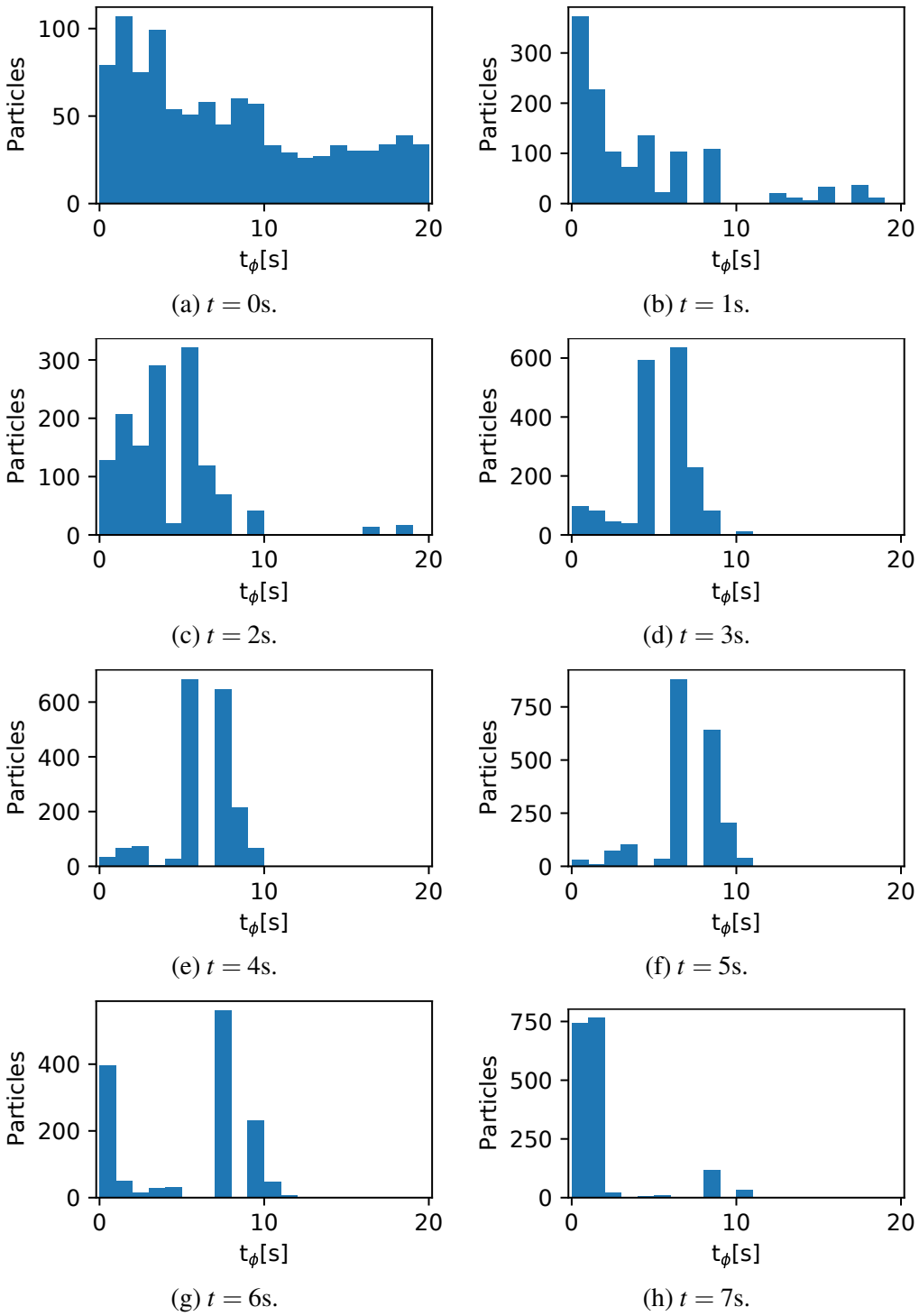
$t = 1$s, the vehicle observes $\phi = YELLOW$ instead of $\phi = GREEN$, making the yellow phase to be more likely in the belief. However, with subsequent correct observations, the belief converges to the true ϕ. The robustness of the estimation can be seen at $t = 6$s, when the vehicle observes $\phi = RED$ instead of $\phi = YELLOW$. Since the transition from green to red is not possible, the model is able to recognize this is an erroneous measurement. Also, after many green observations, the belief starts to change from green to yellow according to Eq. (3.4), since the ego-vehicle has been observing $\phi = GREEN$ for a long period. After the yellow phase is observed, the belief converges once again to the true state.

Fig. 14 shows the estimation of t_ϕ to the same true observations seen in Fig. 13. At $t = 1$s, the particles concentrate around $t_\phi = 0$s, since it is likely that a yellow phase has been initiated according to the erroneous observation. After the subsequent correct observations, the belief changes according to the evolution of t_ϕ (Eq. (3.3)). At $t = 7$s, the particles concentrate in two different regions: around $t_\phi = 8$s (green phase) and $t_\phi = 0$s (yellow phase). At $t = 8$s, the belief about ϕ converges to the yellow phase and the belief about t_ϕ concentrates around $t = 0$s.

3.4.2 Optimal Policy Computation

To evaluate the efficiency of the proposed approach, the experiments are conducted in two different scenarios depending on the values of ϕ and t_ϕ. Also, in each scenario, the perception system's accuracy p is changed in order to analyze robustness. When $p = 1$, no uncertainty in

Figure 14 – Signal timing estimation with respect to time. As more measurements are gathered by the ego-vehicle, it becomes more confident about the true value of t_ϕ.



(a) $t = 0$s. (b) $t = 1$s.

(c) $t = 2$s. (d) $t = 3$s.

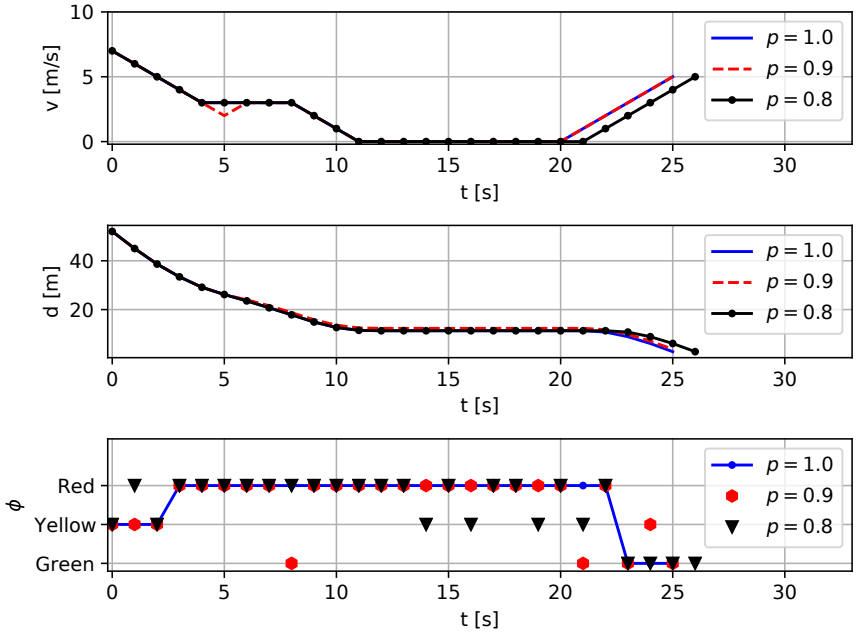
(e) $t = 4$s. (f) $t = 5$s.

(g) $t = 6$s. (h) $t = 7$s.

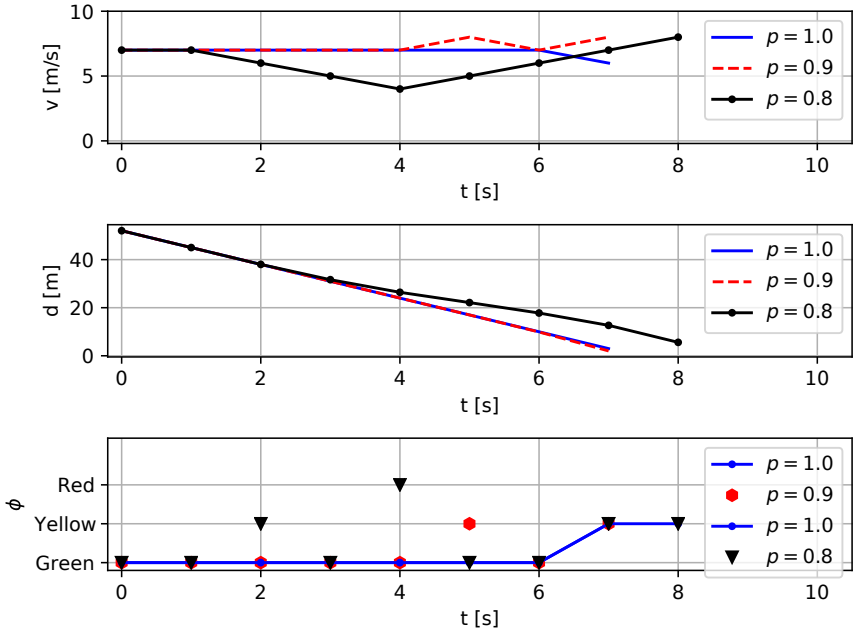
Source: Elaborated by the author.

considered, meaning that the vehicle has full knowledge about ϕ. In all experiments, the vehicle's initial speed is 7.0 m/s, $V_{\text{travel}} = 7.0$ m/s, $a_{\text{lon}} = \{-1, 0, 1\}$ m/s^2 and the maximum allowed speed is 8.33 m/s.

Figure 15 – Planned speed with respect to d and color observations.



(a) Scenario 1.



(b) Scenario 2.

Source: Elaborated by the author.

In the first scenario, the traffic sign is yellow and three seconds remain until the next transition. As it can be seen in Fig. 15a, the vehicle decreases its speed as soon as it receives the first observation. This can be justified by the fact that a yellow-light observation is unlikely when compared to other colors. This makes the belief to converge fastly to the true state of the traffic light, which allows POMCP to compute reliable actions. Furthermore, the computed policy is very similar for all values of p, showing the robustness of the proposed model.

The second scenario is more challenging, since it considers that the traffic light is green and seven seconds remain until the sign turns yellow. Fig. 15b shows the computed policy in this situation. Although the policy is similar for both $p = 1.0$ and $p = 0.9$, when $p = 0.8$ it is quite different. With low uncertainty, the vehicle can quickly estimate the true traffic light sign and try to cross the intersection before the sign turns red. However, with high uncertainty, the observation model cannot give a confident belief about the sign, which makes the vehicle adopt a conservative policy in order not to cross the red-light. As it approaches the intersection, enough observations are gathered in order to make the belief more reliable. This leads the vehicle to try cross the intersection before the lights turn red.

3.4.3 Quantitative analysis

This section presents quantitative results in order to analyze the influence of uncertain measurement awareness. The proposed POMDP model is compared to other scenarios in which the ego-vehicle assumes that the perception system is able to provide perfect measurements. When partially observability is not considered, the problem is reduced to an MDP. Moreover, the influence of a_{lon}, which directly affects comfort, is considered. For both scenarios, the perception system has 80% of accuracy ($p = 0.8$).

Tables 4 and 5 shows the results of POMDP and MPD models for 100 different trajectories, in which the initial ϕ and t_ϕ is randomly chosen. However, in order to conduct a fair comparison, the 100 different initial conditions are the same for both POMDP and MDP models. Also, three features are chosen for this experiment: the mean jerk at run, the number of fails (running red lights), and the mean time to run the traffic light in successful trajectories. The ego-vehicle's initial speed and V_{travel} is kept in 7 m/s as in Section 3.4.2.

Important conclusions can be made about the chosen model and speed rate change range. Regarding a_{lon}, there clearly is a trade-off between comfort and success rate. When the ego-vehicle is able to perform greater deceleration, it can better manage the speed so it does not run the red light, which in turn causes an increase in the mean jerk. Nevertheless, the POMDP model achieves a much lower jerk for $a_{\text{lon}} = \{-1, 0, 1\}$ m/s^2 when compared to the MDP model. In fact, the MDP model presents similar jerk in both scenarios. Since it does not account for

Table 4 – Results for $a_{\text{lon}} = \{-1, 0, 1\}$ m/s^2.

	Mean jerk (m/s^3)	Fails	Mean time (s)
POMDP	**0.38**	**14**	15.8
MDP	0.59	32	**14.4**

Table 5 – Results for $a_{\text{lon}} = \{-2, 0, 1\}$ m/s^2.

	Mean jerk (m/s^3)	Fails	Mean time (s)
POMDP	**0.67**	**3**	17.3
MDP	**0.67**	20	**15.3**

noisy measurements, it changes its speed in a too reactive manner causing larger jerk values.

The POMDP model also presents much better results with respect to the number of fails. The capability of estimating ϕ through successive measurements makes the ego-vehicle aware of possible traffic light color transitions, which can be used to plan better policies so it does not cross red lights. This causes the ego-vehicle to adopt more cautious behaviors, since it tends to take conservative actions before the convergence of ϕ and t_ϕ beliefs, as in Fig 15a. This justifies the larger time taken by the POMDP model to cross the traffic light.

3.5 Final Considerations

This chapter presents a POMDP model for autonomous vehicle decision-making at signalized intersections in the presence of uncertainty and limited speed rate changes. Results show that the proposed approach provides reliable results even in the presence of highly uncertain observations when compered to approaches that assume perfect observations. Next chapter covers the problem of decision-making in the presence of other vehicles, which is a common scenario in real applications.

CHAPTER

4

INTERACTION-AWARE DECISION-MAKING ON MULTI-LANE ROADS FOR AUTONOMOUS DRIVING

The presence of autonomous vehicles on public roads is getting closer and closer becoming reality. To achieve that, they are expected to be able to deal with surrounding vehicles on multi-lane roads. Hence, being aware of surrounding vehicles' motion is essential to provide a fluid navigation in this scenario. However, if communication between the vehicles is not available, the motion inference must be made only from sensors information. This chapter presents a decision-making framework using Partially Observable Markov Decision Process (POMDP) for autonomous driving on multi-lane roads in the presence of sensor noisy measurements and other vehicles as well The current reference lane of a surrounding vehicle is assumed partially observable in order to account for the possibility of lane changes. Therefore, an ongoing lane change can be inferred from a sequence of observations. Also, the POMDP model considers that a vehicle can influence on the motion of other vehicles and vice-versa, which can significantly benefit the autonomous vehicle in taking decisions. Results from simulation show the efficiency of the proposed decision-making framework by considering the execution of regular maneuvers, such as merging into the gap of two other vehicles and reacting to a sudden lane change of a vehicle in front.

4.1 Introduction

One key feature of autonomous vehicles is the capability of making complex decisions in urban roads, since it is a highly stochastic domain due to other traffic participants. This requires autonomous cars to be aware of surrounding vehicles' movements in order to anticipate to their maneuvers, avoiding behaving in over-reactive manner. Moreover, it is expected that autonomous vehicles act in accordance to some level of social acceptability, which means not only obeying

traffic rules and avoiding accidents, but also behaving in such way to provide a smoother joint navigation among all traffic participants. This includes being aware of and giving way to other vehicles when necessary, avoiding aggressive overtaking, keeping the traffic flow, etc. This challenging task requires being able to reason about future actions of all traffic participants, including the autonomous vehicle itself.

When navigating on roads with multiple lanes, the future actions of surrounding vehicles can be summarized in executing lane changes and changing speed. However, the autonomous vehicle are not able to know when each vehicle will take action, unless there is direct communication between them (CHEN; ENGLUND, 2015; BEVLY *et al.*, 2016). In general, surrounding vehicles' intentions can only be estimated through the automated vehicle's perception system, which gives the pose and velocity of moving obstacles. Therefore, surrounding vehicles' intention can be assumed as partially observed, since it can be inferred from the available information provided by sensors. Nevertheless, this intention estimation comes along with inevitable uncertainty, which cannot be neglected in the decision-making process when safe, reliable navigation is pursued.

This chapter presents a decision-making framework for autonomous vehicles when navigating on multi-lane roads in the presence of other vehicles. Partially Observable Markov Decision Process (POMDP) is used in order to address the issue of intention estimation of surrounding vehicles. The solution of the POMDP is an optimal policy that are composed by a lateral action (making a lane change or staying on the current lane) and a longitudinal action (breaking, accelerating or keeping the current speed). The POMDP problem is solved by using the online POMDP solver Partially Observable Monte Carlo Planning (POMCP) (SILVER; VENESS, 2010), which can handle discrete and continuous states. The main contributions of this work are:

- An interaction-aware motion model that is capable of dealing with future interaction between surrounding vehicles and the autonomous vehicle itself.
- A centralized action model that is used to compute lateral and longitudinal actions.
- An observation model that can estimate an ongoing lane change only by making use of the perception system, without the need of communication between vehicles.

The remainder of this work is organized as follows: Section 4.2 covers related works; the formal definition and scope of the problem is discussed in Section 4.3; the proposed POMDP model is detailed in Section 4.4; in Section 4.5 simulation results are presented and discussed; and, finally, remarks and future work are presented in Section 4.6.

4.2 Related Work

This section covers relevant work related to decision-making for autonomous vehicles in multi-lane roadways, discussing the main approaches that have been applied in this field.

4.2.1 Finite State Machines

In the Urban Darpa Challenge (DCU), Finite State Machines (FSM) was the approach used by the two winners of the competition: CMU's Boss (URMSON *et al.*, 2008) and Stanford's Junior (MONTEMERLO *et al.*, 2008). In this context, the action chosen by the controlled autonomous vehicle (referred to as *ego-vehicle* from here on out) depends on its current state. The states are high level abstractions of the current situation experienced by the vehicle. Instead of using only one FSM, Montemerlo *et al.* (2008) propose a Hierarchical State Machine (HSM), in which each state represents a *context* and each context is described by a FSM. The three proposed contexts are *Lane Driving*, *Intersection Handling* and *Goal Selection*. Two states in the *Lane Driving* context are directly related to lane changes, which requires the vehicle to make lateral motion: LANE_SELECTOR and MERGE_PLANNER. The ego-vehicle chooses between staying on the current lane or changing to the adjacent one after evaluating some metrics, such as the distance to the vehicle in front, its speed and a safe minimal gap to execute the maneuver. Despite being capable of handling the scenarios in DCU, such as intersection negotiation, merging and lane change, these approaches require engineering the possible states the vehicle might be in, which can become a difficult task as more complex scenarios are considered. Moreover, those approaches does not consider uncertainty during the planning phase.

4.2.2 Planning under uncertainty

The driving task requires being able to reason about the future actions of the ego-vehicle. In some works, surrounding vehicles are either considered to be static obstacles (OBAYASHI; UTO; TAKANO, 2016) or to follow a constant speed during planning (MOUHAGIR *et al.*, 2017). These assumptions can lead to reactive planning since the motion prediction of other vehicles are not considered. Estimating vehicles' maneuvers can be straightforward if they can communicate with each other. In the literature, this is referred to as Vehicle to Vehicle (V2V) communication (KATRAKAZAS *et al.*, 2015). Nevertheless, the planner cannot rely only in perfect information, since it cannot be guaranteed that it will be available in all circumstances (HUBMANN *et al.*, 2018a). When this is the case, the vehicles' maneuver can be inferred through *motion prediction models*. Lefèvre, Vasquez and Laugier (2014) group the behavior prediction into three classes depending on the level of abstraction:

- *Physics-based* motion models: simplest models, based on laws of physics governing vehicles' motion.

- *Maneuver-based* motion models: consider, besides laws of physics, the road network topology and traffic rules to improve prediction.

- *Interaction-aware* motion models: take the interaction between vehicles into account, which leads to inter-dependencies between vehicles' motion.

According to Lefèvre, Vasquez and Laugier (2014), *interaction-aware* motion models lead to better estimation, since they consider the influence of one vehicle in other vehicle's motion. However, uncertainty is inherent in such framework, since the prediction is made relying only on environment information gathered by sensors.

Partially Observable Markov Decision Process (POMDP) is commonly applied to solve planning problems for autonomous vehicles in the presence of uncertainty (SCHWARTING; ALONSO-MORA; RUS, 2018). In this framework, vehicles' *intentions* are considered as partially observable states. Whereas Markov Decision Process (MDP) aims to compute an optimal *policy* that maps states into actions (RUSSELL, 2010), since the state space is fully observable, POMDP calculates a policy that maps *beliefs* into actions. A *belief* is a probability distribution over the state space (KAELBLING; LITTMAN; CASSANDRA, 1998). The desired ego-vehicle's behavior is specified by the maximization of a *reward function*. POMDP has been used in many areas of research regarding decision-making in autonomous driving: to deal with unknown intention of pedestrians (BANDYOPADHYAY *et al.*, 2013); to handle situations at T-junction negotiations in the presence of occlusions (BRECHTEL; GINDELE; DILLMANN, 2014); intersection negotiation with surrounding vehicles (HUBMANN *et al.*, 2018a)(GINDELE; BRECHTEL; DILLMANN, 2015) (SONG; XIONG; CHEN, 2016); and merging onto a road or into a roundabout (LIU *et al.*, 2015).

Specifically in lane change scenarios, Ulbrich and Maurer (2013) propose a discrete POMDP model to decide when to initiate the maneuver. A Signal Processing Network (SPN) computes the probability of safely executing the lane change and whether it is beneficial for the ego-vehicle. Ulbrich and Maurer (2015) incorporate the uncertain SPN in the measurement model required to estimate unknown states in POMDP, such as the maneuver safety or the gap quality in which the ego-vehicle is going to merge. The POMDP model itself is modified to accommodate mixed-integer states, such as the distance to vehicles in front. Sezer (2018) present a discrete mixed-observable MDP in order to account for uncertainty in the transition and observation model. However, in these works, the POMDP model does not explicitly define the interaction between the vehicles. Hubmann *et al.* (2018b) propose a POMDP model to planning speed and lane changes maneuver in heavy traffic conditions. Despite of estimating whether surrounding vehicles intend to give the right of way to the ego-vehicle merge in a target lane, they do not consider that surrounding vehicles are able to perform lane changes. On the other hand, Sunberg, Ho and Kochenderfer (2017) assumes that other vehicles can change lane in the planning phase, however they only present quantitative results by comparing them to other

deterministic planners.

In contrast to previous works, this chapter proposes an unified POMDP model, in which the optimal policy is composed of lateral and longitudinal actions. Also, the proposed POMDP model is able to account for more complex behaviors of other traffic participants, such as when they execute lane changes or reduce speed in order to give way to other vehicles. Moreover, the quality of the belief estimation is detailed in order to justify the use of a POMDP model to handle the problem discussed in this chapter.

4.3 Problem Statement

As discussed before, the goal of the proposed decision-making framework is the navigation of the ego-vehicle in multi-lane roads where other vehicles may be present. Therefore, the focus is the development of a *behavior planner*, which is responsible for computing tactical decisions, such as changing lanes and adjust the ego-vehicle' speed, in order to avoid collision and to keep the velocity around the reference value given by the highest-layer in the decision-making hierarchy based on traffic rules (Fig 1). The perception system, trajectory planner and controllers are assumed to be available as well as the lane paths constituting the road network.

Fig. 16 depicts the problem considered in this work. The ego-vehicle is navigating in a road constituted of I lanes, where $r^{(i)}\ \forall i \in I$ stands for a specific path that leads to lane i, with the right-most lane represented by $i = 0$. All K vehicles in the scene, including the ego-vehicle, can chose between changing to an adjacent lane or staying in its current one. Because of the lack of V2V, the ego-vehicle is unable to determine the current path of surrounding vehicles, since all possible paths initiate from the same point. Therefore, the ego-vehicle can only infer if a lane changing is in course throughout subsequent measurements coming from the perception system. Also, all K vehicles are assumed to be able to adjust their speed in order to interact with the other traffic participants. While the intention of surround vehicles can be only inferred, the ego-vehicle's actions are determined by a policy, which is the result of solving the POMDP problem.

4.4 POMDP Model

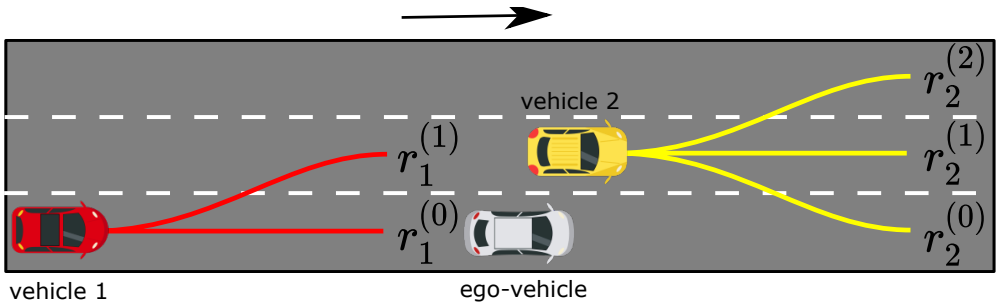
This section presents the POMDP model used to compute near-optimal actions for the ego-vehicle when it is navigation in a multi-lane road in the presence of other vehicles.

4.4.1 State Space

The state space $\mathcal{S}$ includes the states of all vehicles considered in the scene,

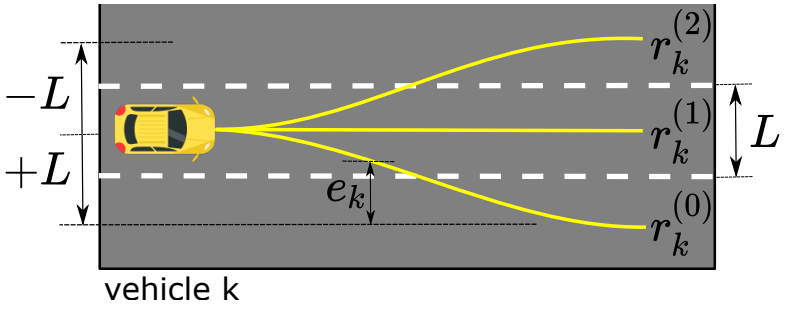
$$\mathcal{S} = [s_0 \quad s_1 \quad s_2 \quad \cdots \quad s_K]^{\mathrm{T}}, \tag{4.1}$$

Figure 16 – Problem statement. The ego-vehicle navigates on a multi-lane road with the presence of other vehicles. These vehicles are following unknown paths given by $r^{(i)}$.



Source: Elaborated by the author.

Figure 17 – Vehicle k is following one of the unknown paths $r_k^{(i)}$. Each path leads to a different lane with width L, where e_k is the lateral deviation from the reference lane.



Source: Elaborated by the author.

where s_0 represents the state of the ego-vehicle and $s_k \in \{1, \cdots, K\}$ are the states of surrounding vehicles.

The ego-vehicle's state s_0 is defined as

$$s_0 = [d_0 \quad v_0 \quad r_0]^{\mathrm{T}}, \tag{4.2}$$

where d_0, v_0 and r_0 are: the distance from the ego-vehicle center to the initial point of the road, the speed, and the followed path, respectively. Similarly, surrounding vehicles' states are defined as

$$s_k = [d_k \quad v_k \quad r_k \quad e_k]^{\mathrm{T}}. \tag{4.3}$$

The route r_k is one of the possible paths that the vehicle can follow from its current position, as demonstrated in Fig. 17. Since POMCP is particle-based, the lateral deviation from the reference lane e_k is added in order to determine the position the vehicle occupies on the road given r_k. Hence, the ego-vehicle can be aware of possible lane transitions performed by surrounding vehicles.

4.4.2 Action Space and Transition Model

The transition on the state variables related to the vehicles' longitudinal motion is constrained to the following motion model:

$$\begin{bmatrix} d'_k \\ v'_k \end{bmatrix} = \begin{bmatrix} d_k \\ v_k \end{bmatrix} + \begin{bmatrix} v_k \\ a_{\text{lon}k} \end{bmatrix} \Delta t + \frac{1}{2} \begin{bmatrix} a_{\text{lon}k} \\ 0 \end{bmatrix} \Delta t^2, \tag{4.4}$$

in which $a_{\text{lon}k}$ is the speed rate change of vehicle k, and Δt is the time step. Also, the vehicles can perform lane changes according to

$$r_k^{\prime(i')} = r_k^{(i + a_{\text{lat}k})}. \tag{4.5}$$

The ego-vehicle is able to chose between three longitudinal actions and three lateral actions:

$$\mathcal{A}_{\text{lon}} = \{Brake \quad StayConst \quad Accelerate\},$$
$$\mathcal{A}_{\text{lat}} = \{LLC \quad SCL \quad RLC\},$$

where *Brake*, *StayConst* and *Accelerate* stand for a negative, a null and a positive speed rate change, respectively. For lateral maneuvers, *LLC*, *SCL* and *RLC* represent a change to the adjacent left lane, staying in the current lane and a change to the adjacent right lane, respectively. In order to compact the action space, $\mathcal{A}_{\text{lon}}$ and $\mathcal{A}_{\text{lat}}$ are grouped as $\{Brake, SCL\}$, $\{Accelerate, SCL\}$, $\{StayConst, LLC\}$, $\{StayCon, RCL\}$, $\{StayConst, SCL\}$, summing up to five different actions instead of the nine possible combinations between $\mathcal{A}_{\text{lon}}$ and $\mathcal{A}_{\text{lat}}$. Thus, the ego-vehicle chooses the best action according to the optimal policy. The values of the longitudinal actions are shown in Table 6.

While the ego vehicle's motion is determined by the optimal policy, surrounding vehicles' actions must be predicted, since V2V communication is not considered. These actions are inferred from the vehicle's position on the road and velocity, and also from the interaction with other vehicles, including the ego-vehicle.

The vehicles are assumed to perform lane changes in order to a) overtake a slower vehicle, b) to return to its original lane or even c) to reach an exit from the road. Instead of predicting such maneuvers, $a_{\text{lat}k}$ is chosen according to the following stochastic model:

$$P_{lat} = \{LLC = 0.05 \quad SCL = 0.9 \quad RLC = 0.05\}, \tag{4.6}$$

where P_{lat} is the probability of choosing a lateral action by considering that lane changes are less likely than staying in the current lane. Whenever a lane change is chosen, e_k is set with respect

Table 6 – Speed rate change.

Brake	*StayConst*	*Accelerate*
-1 m/s	0	1 m/s

to the lane width L (Fig. 17) and changes according to

$$e'_k = e_k + sign(a_{\text{lat}k}) \cdot 0.875\Delta t, \tag{4.7}$$

where $sign(a_{\text{lat}k})$ is negative for *LLC* and positive otherwise, and assuming that surrounding vehicles move 0.875 m/s towards the target lane. Further, a lane change cannot be aborted once it has been initiated, meaning that a new lane change for a given particle is only allowed when $e_k = 0$.

The longitudinal motion of vehicle k is assumed to be determined by the distance D to a vehicle in front:

- $D > 15$ m: In this case, the longitudinal action $a_{\text{lon}k}$ is simply assumed as a Gaussian noise $\mathcal{N}(0, \sigma^2)$, by considering that vehicles tend to keep a nearly-constant velocity when they do not directly interact with other vehicles.

- $D < 15$ m: Now, $a_{\text{lon}k}$ is determined from the chosen $a_{\text{lat}k}$. If the vehicle stays on the current lane, i.e. $a_{\text{lat}k} = SCL$, it must decrease its speed to avoid collision. Vehicle k starts to follow the vehicle in front (assumed as vehicle $k-1$) and adjusts its speed according to

$$a_{\text{lon}k} = max\big(a_{min},\ (v_{k-1} - v_k)/\Delta t + \mathcal{N}(0,\ \sigma^2)\big), \tag{4.8}$$

  where a_{min} is the minimum speed rate change allowed, assumed as -1 m/s^2. Otherwise,

$$a_{\text{lon}k} = max\big(0,\ \mathcal{N}(0,\ \sigma^2)\big), \tag{4.9}$$

  by considering that vehicles do not tend to reduce their speed when performing an overtaking.

4.4.3 Observation Space and Observation Model

The ego-vehicle's perception system gives the position and velocity of all vehicles considered in the scene, in which the position measurement is assumed to be corrupted by a Gaussian noise. However, it cannot determine which path is being followed by surrounding vehicles. Therefore, the ego-vehicle must infer vehicles' reference lanes by gathered observations.

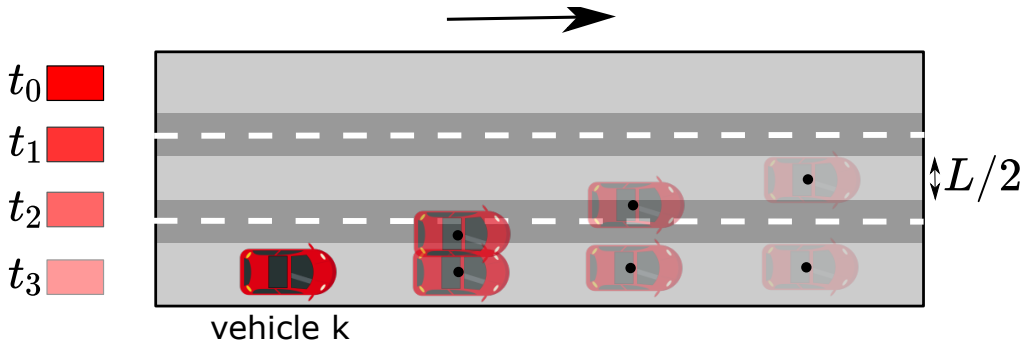
The observation space $\mathcal{O}$ is defined as

$$\mathcal{O} = [o_0 \quad o_1 \quad o_2 \quad ... \quad o_K]^{\text{T}}, \tag{4.10}$$

where o_0 represents the observation of the ego-vehicle and $o_k \in \{1, \cdots, K\}$ are the observation of surrounding vehicles. As discussed before, the ego-vehicle is able to directly observe its own state,

$$o_0 = [d_0 \quad v_0 \quad r_0]^{\text{T}}, \tag{4.11}$$

Figure 18 – Simulated observation given r'_k and e'_k. The road is divided into regions by considering the vehicle close to the lane center (light gray) or between two adjacent lanes (dark gray).



Source: Elaborated by the author.

and the position and velocity of surrounding vehicles,

$$o_k = [x_k \quad y_k \quad v_k]^{\mathrm{T}}, \tag{4.12}$$

where $(x_k,\ y_k)$ is the position of vehicle k in global coordinates.

The longitudinal movement of surrounding vehicles can be predicted with low uncertainty as described in Section 6.3.1.2. However, the same cannot be stated for their lateral movement, since they perform lane changes according to (4.6). Hence, whereas the prediction of v_k relies only on the transition model over time, the observation o_k can be used to infer whether a lane change is occurring, resulting in the prediction of future positions on the road.

POMCP does not require to explicitly define an observation model. Instead, a potential observation must be sampled from the generative model. Fig. 18 shows how future observations are computed. The future position of the vehicle $(x'_k,\ y'_k)$ is simulated given r'_k and e'_k. After that, the continuous observation o_k are transformed into a discrete observation that consider different regions along the road according to Fig. 18. This transformation is required since POMCP assumes discrete observations to expand $\mathcal{T}$. In order to account for uncertainty in the observation, such as deviations from the lane center that do not result in lane changes because of noisy measurements, a Gaussian noise $\mathcal{N}(0, 1.0)$ is added to the simulated observation.

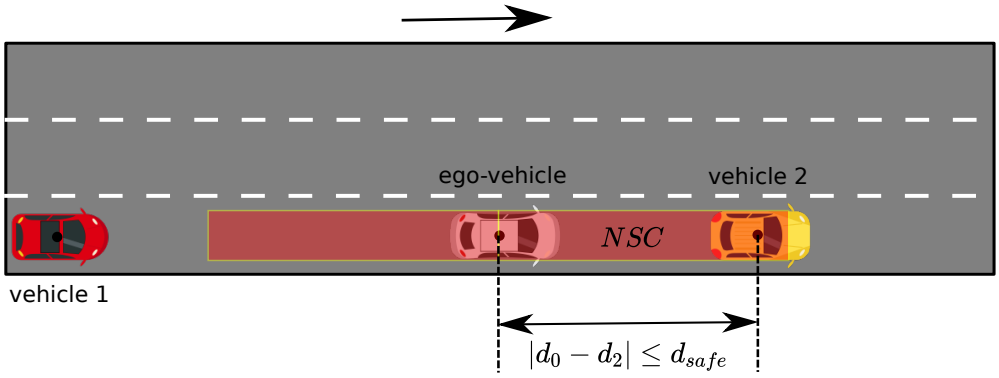
4.4.4 Reward Model

The ego-vehicle must navigate while 1) avoiding deviating from its reference speed and 2) avoiding collisions. The reward function is defined as

$$R(s,\ a) = \begin{cases} 0, & \text{if } NSC, \\ R_v + R_{a_{\text{lon}}} + R_{a_{\text{lat}}} + R_r, & \text{otherwise}, \end{cases} \tag{4.13}$$

where NSC means that the ego-vehicle is not in a safe condition (Fig. 19). In order to be considered in NSC, the ego-vehicle must be, at least, 3 m from vehicles in which $r_k = r_0$. The

Figure 19 – Definition of *NSC* in the reward model.



Source: Elaborated by the author.

distance of 3 m is applied in order to ensure a safe margin for lane changes performed by the ego-vehicle. In this work, the length of all vehicles is assumed as 2.5 m, which means that the ego-vehicle is in *NSC* when $d_{safe} \leq 5.5$ m. However, the vehicles' length can be given by the perception system and can easily be incorporated into the model. Since being in *NSC* is not desirable, the total reward is null when this condition is met. Otherwise, the reward is given by a sum of terms, where R_v gives the reward of keeping the velocity around the reference speed and is computed as:

$$\begin{aligned} R_v &= (1/v_{ref})^2, \qquad v_0 \leq v_{ref} \\ R_v &= 2 - 1/v_{ref}, \qquad v_0 > v_{ref}. \end{aligned} \tag{4.14}$$

where v_{ref} is the reference speed given by the mission planner. When $v_0 \leq v_{ref}$, R_v is squared in order to decrease the reward for low speeds (it should be noted that $R_v \leq 1$). This prevents the ego-vehicle to decrease its speed to follow a slower vehicle. $R_{a_{\text{lon}}} = -0.1$, $R_{a_{\text{lat}}} = -0.25$ are punishments for changing speed and changing lanes, respectively. This forces the ego-vehicle to not execute unnecessary actions. The punishment term R_r is used to keep the vehicle on a predefined lane given by the mission planner, in a similar way as for the velocity. Thus, $R_r = 0$ for the predefined lane and decreases by an amount of 0.1 for more distant lanes.

4.5 Results

Conducting experiments in real traffic environments with several vehicles is a very complex task. An alternative to that is the use of simulators dedicated to autonomous driving research. The simulated environment must correspond, in a satisfactorily manner, to that encountered in real traffic. *CARLA* (Car Learning to Act) simulator (DOSOVITSKIY *et al.*, 2017) is used as the simulation platform for experimental validation of the proposed approach. The system containing the simulation environment and algorithms runs on a Intel Core i7-7700 CPU with 3.60 Hz, a 15.5 GB memory RAM and a NVIDIA GeForce GTX 1050 Ti GPU. Both POMDP model and POMCP are implemented in *Python*, which provides a simpler integration to CARLA.

Table 7 – POMCP parameters.

Δt	c	γ
1.5 s	3.0	0.99

The experiments are conducted considering a road with four lanes. The ego-vehicle is the only controlled vehicle. The goal of the proposed decision-making is to compute the reference speed and lane to be followed by the ego-vehicle at each time step. The predefined lane, which is given by the mission planner, is the right-most lane. The trajectory planner and control modules required to execute the computed decisions are assumed already available, as well as the perception system.

The main parameters of POMCP are depicted in Table 7. As it should be noted, the time step Δt imposes a trade-off between prediction horizon and re-planning frequency. $\Delta t = 1.5$ s is chosen in order to reach both requirements satisfactorily. The parameter $c = 3$ allows enough exploration in the belief tree and maintains the prediction horizon around 6 s, an acceptable value in autonomous driving (LEFÈVRE; VASQUEZ; LAUGIER, 2014). The prior probabilities of surrounding vehicles' path is assumed as 0.8 for the closest lane to the vehicle and 0.1 to the adjacent ones.

To show the attributes of the proposed approach, the experiments are divided into three scenarios, where the position configurations and velocities of the surrounding vehicles differs in each scenario. In all experiments, the reference speed given by the mission planner is 7.0 m/s.

4.5.1 Experiment 1: Surrounding vehicles' motion prediction

The first set of experiments considers two surrounding vehicles, as in Fig. 20, and aims to show how the interaction motion model is applied in order to predict $a_{\text{lon}k}$. Table 8 shows the observed values when the planner is started. Considering this scenario, two situations can arise: vehicle 1 overtakes vehicle 2, or vehicle 1 does not overtake vehicle 2 and gives way to the ego-vehicle. Both scenarios are analyzed in detail.

4.5.1.1 Overtaking

Fig. 21a shows the current belief about vehicle 1's path. At $t = 3$ s, the ego-vehicle observes vehicle 1 in the region between lane 0 and lane 1. At this point, the ego vehicle becomes aware of the ongoing lane change. In the next time step ($t = 4.5$ s), the current belief finally

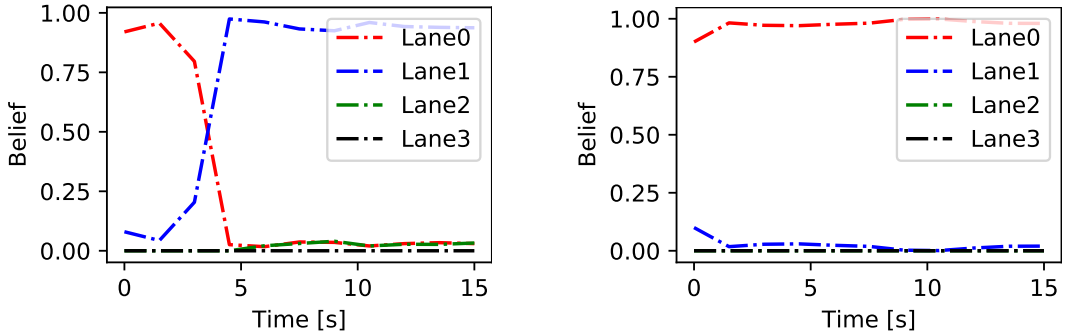
Table 8 – Variables of Experiment 1.

d_0(m)	d_1(m)	v_1(m/s)	d_2(m)	v_2(m/s)
113.1	127.2	6.5	151.1	4.1

Figure 20 – Experiment 1: surrounding vehicles' motion prediction.

Source: Elaborated by the author.

Figure 21 – Estimation of r_1 over time in Experiment 1.

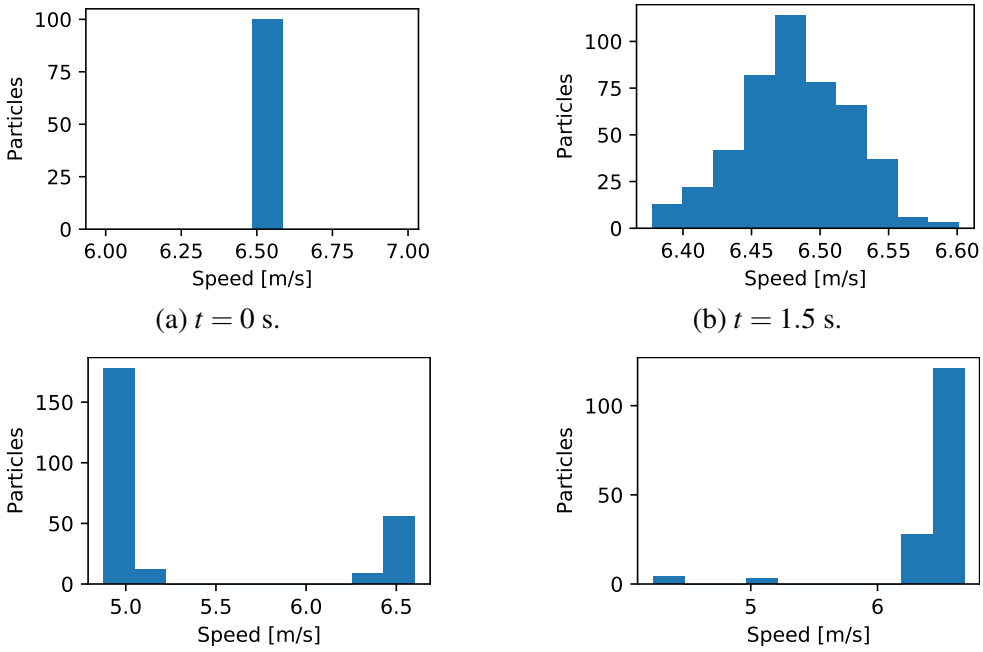


(a) Overtaking. (b) Giving way.

Source: Elaborated by the author.

converges to lane 1. The belief about the chosen lane directly influences the predicted speed of vehicle 1. Fig. 22 depicts the number of particles with respect to the predicted speed. As it should be noted, all particles have the same speed value at the beginning, since they are originated from the true observed value. In the next time step, the particles assume different values because of the noise uncertainty σ added to the transition model. At $t = 3.0$ s, vehicle 1 is in the region between lane 0 and lane 1, and approximates to vehicle 2, which makes the particles concentrate in two regions. The particles around 5.0 m/s consider that vehicle 1 is sufficiently close to vehicle 2, forcing a speed reduction. On the other hand, some particles consider an ongoing lane change, as shown in Fig. 21a, and are concentrated around 6.5 m/s. This concentration in two different regions is due to the uncertainty considered in the observation model. After confirmation of the lane change at $t = 4.5$ s, the particles converge to $v_1 = 6.5$ m/s.

Figure 22 – Prediction of v_1 in the overtaking scenario.

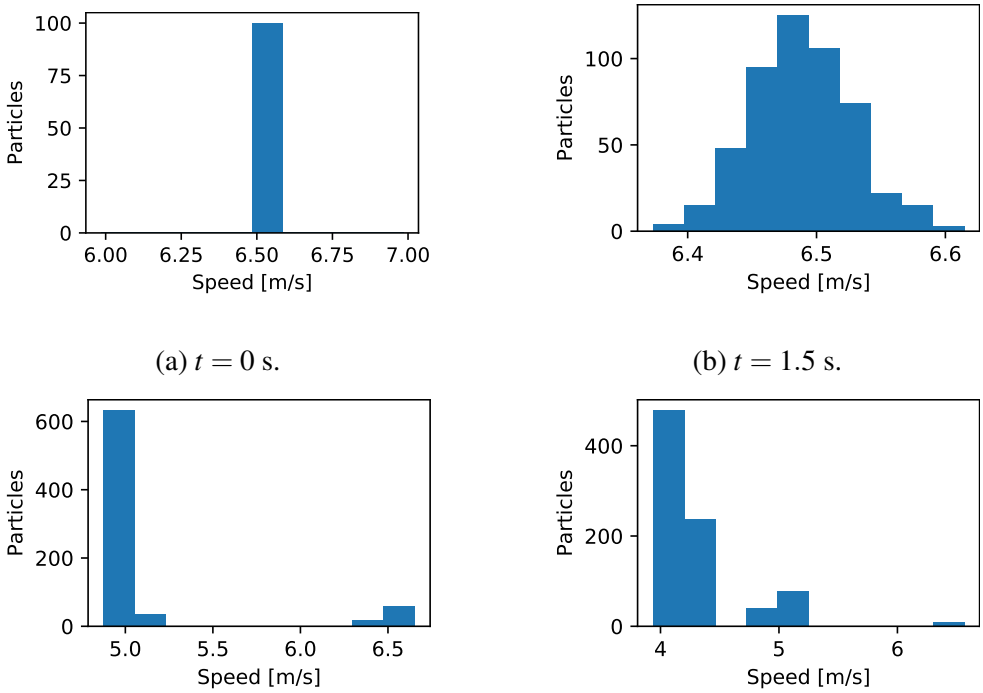


(a) $t = 0$ s. (b) $t = 1.5$ s.

(c) $t = 3.0$ s. (d) $t = 4.5$ s.

Source: Elaborated by the author.

Figure 23 – Prediction of v_1 in the giving way scenario.



(a) $t = 0$ s. (b) $t = 1.5$ s.

(c) $t = 3.0$ s. (d) $t = 4.5$ s.

Source: Elaborated by the author.

4.5.1.2 Giving way

In this scenario, vehicle 1 does not perform a lane change, as shown in Fig. 21b, which forces it to adapt its speed according to vehicle 2. Fig. 23 depicts how the velocity is predicted in this situation. Up to 3.0 s, the analysis is very similar to the one made for Fig. 22. However, vehicle 1 is observed close to the line center of lane 0 at $t = 4.5$ s, meaning that a lane change is not happening. This makes the particles concentrate around $v_1 = 4.1$ m/s, which is the speed of vehicle 2.

4.5.2 Experiment 2: Lane merging

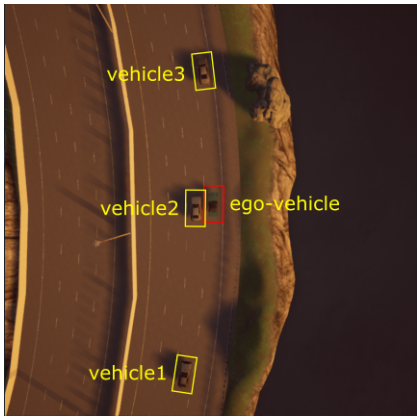
An interaction-aware motion model can be required in order to achieve more complex maneuvers, as presented in Fig. 24. Table 9 describes the state values at the beginning of the planning. As it can be seen, the ego-vehicle must chose between merging into the gap between vehicle 1 and vehicle 2, or decreasing its speed in order to follow vehicle 3 until being completely overtaken by vehicle 1.

Fig. 25 depicts the resulting policy with respect to time, while 26 demonstrates the evolution of the vehicles' position on the road. For brevity, only some time steps are chosen to be shown. As it can be noted, v_0 is decreased to 5.5 m/s as the ego-vehicle approximates to vehicle 3 in order to wait for the gap between vehicle 1 and vehicle 2. At $t = 6.0$ s, v_0 is increased to 7.0 m/s, which indicates that the ego-vehicle is ready to perform a lane change. Once a safe distance from the surrounding vehicles is available at $t = 7.5$ s, the ego-vehicle changes lane and stays between vehicle 1 and vehicle 2 until it completely overtakes vehicle 3, returning to lane 0 at $t = 16.5$ s.

Table 9 – Variables of experiment 2.

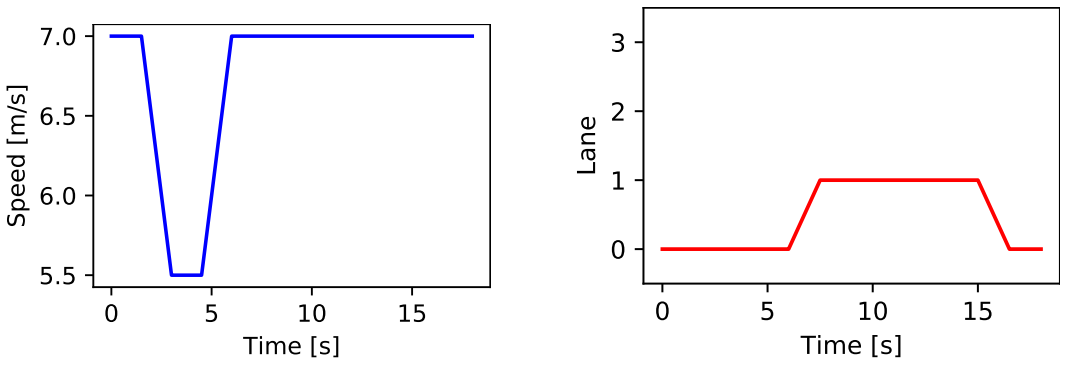
d_0(m)	d_1(m)	v_1(m/s)	d_2(m)	v_2(m/s)	d_3(m)	v_3(m/s)
101.1	69.6	8.0	100.7	7.7	122.9	4.8

Figure 24 – Experiment 2: lane merging.



Source: Elaborated by the author.

Figure 25 – Resulting policy in Experiment 2. The ego-vehicle decreases its speed and merges into the gap between vehicle 1 and vehicle 2 in order to overtake vehicle 3.



(a) Speed. (b) Lane.

Source: Elaborated by the author.

Figure 26 – Sequence of frames of Experiment 2.

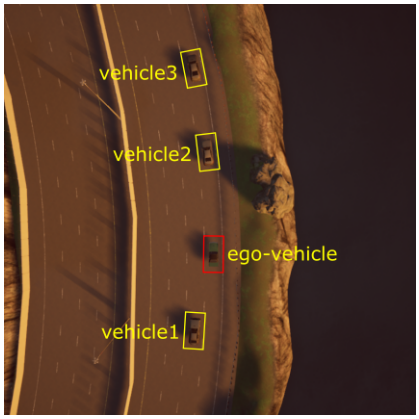
(a) $t = 3.0$ s. (b) $t = 6.0$ s. (c) $t = 9.0$ s.

(d) $t = 12.0$ s. (e) $t = 15.0$ s.

Source: Elaborated by the author.

This complex maneuver is only possible because the ego-vehicle is aware of vehicle 1's motion. According to the interaction-aware motion model, since vehicle 1 decided to stay on

Figure 27 – Experiment 3: reacting to lane changes.



Source: Elaborated by the author.

Table 10 – Variables of Experiment 3.

d_0(m)	d_1(m)	v_1(m/s)	d_2(m)	v_2(m/s)	d_3(m)	v_3(m/s)
108.8	94.6	7.4	125.0	5.1	140.3	4.0

lane 1 after the ego-vehicle changes lane, it must decrease its speed in order to avoid colliding with the ego-vehicle. As it can be seen in Fig. 26, vehicle 1 decreases its speed, with v_1 around 7.0 m/s, and starts to follow the ego-vehicle.

4.5.3 Experiment 3: Reacting to lane changes

As discussed in Section 4.5.2, the ego-vehicle can benefit from being aware of other vehicles' longitudinal motion. Nevertheless, being aware of their lateral motion also plays a key role during planning. Fig. 27 and Table 10 illustrate this situation. The ego-vehicle approximates vehicle 2, which is close to vehicle 3. Therefore, both the ego-vehicle and vehicle 2 have slower vehicles in front of them.

Fig. 28 and 29 show the optimal policy resulting from the online POMDP solver. At $t = 1.5$ s, the ego-vehicle merges into lane 1 to overtake vehicle 2. However, the same action is taken by vehicle 2 in order to overtake vehicle 3. As vehicle 2 moves towards lane 1, the belief about its current path begins to change, similar to Fig. 21a. The ego-vehicle predicts the lane change and decides to merge to lane 2 in order to overtake both vehicle 2 and vehicle 3.

Despite being right behind vehicle 2 at the moment it executes the lane change, the ego-vehicle is able to avoid a collision due to the prediction about the future lane that vehicle 2 is going to occupy. Since POMCP can simulate not only future states as well as future observations, the node in $\mathcal{T}$ representing the belief in which vehicle 2 changes lane is expanded during the policy computation. When this belief is reached after the observation, a plan that considers vehicle 2's motion is already available. This demonstrates that the POMDP model can deal with uncertain lane changes performed by surrounding vehicles.

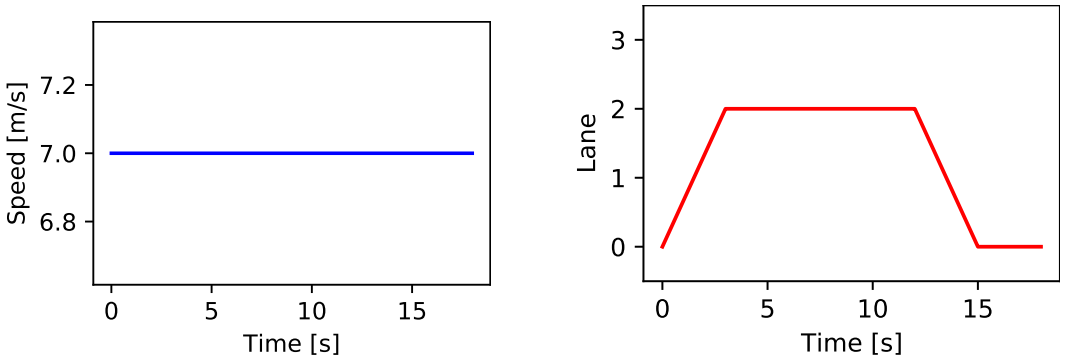
Figure 28 – Sequence of frames of Experiment 3.

(a) $t = 3.0$ s. (b) $t = 6.0$ s. (c) $t = 9.0$ s.

(d) $t = 12.0$ s. (e) $t = 15.0$ s.

Source: Elaborated by the author.

Figure 29 – Resulting policy in Experiment 3. The ego-vehicle properly reacts to an ongoing lane change performed by a vehicle in front.



(a) Speed. (b) Lane.

Source: Elaborated by the author.

4.6 Final Considerations

This chapter presents a POMDP model for solving decision-making problems on multi-lane roads. The main contribution is a POMDP model, which considers interaction between vehicles as well as the possibility of lane changes executed by surrounding vehicles. The POMDP model does not require V2V communications and is capable of estimating surrounding vehicles' actions only through sensor information. The results demonstrates that the proposed model can handle complex situations encountered in multi-lane roads, such as merging into the gap between two vehicles and reacting to sudden lane changes performed by a vehicle in front.

Tuning a reward function with many features, such as the one proposed in this work, has proven to be a difficult task, since it must encode antagonistic behaviors, such as deviating from slower vehicles and avoiding to change lane. This problem is addressed in the next chapter.

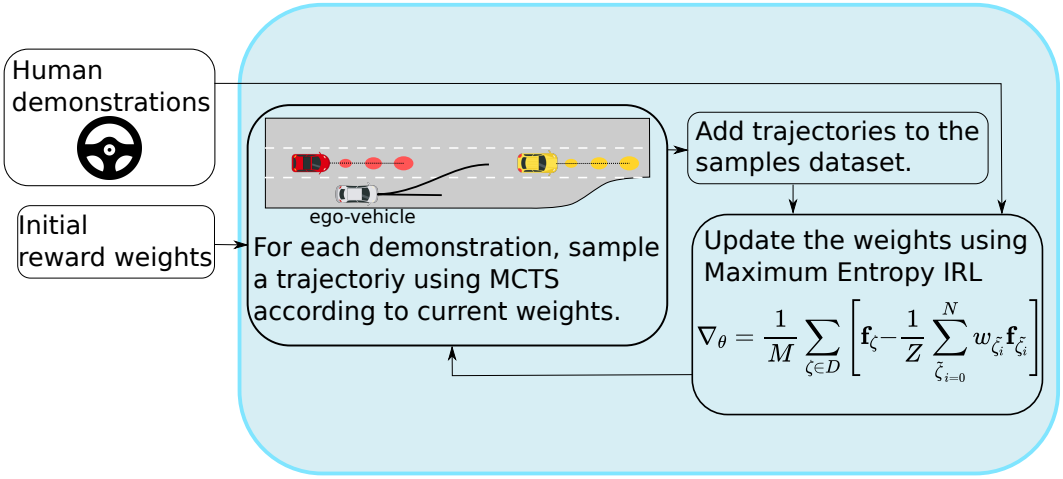
CHAPTER

5

MAXIMUM ENTROPY INVERSE REINFORCEMENT LEARNING USING MONTE CARLO TREE SEARCH FOR AUTONOMOUS DRIVING

Autonomous vehicles must be capable of driving safely and having some level of social compliance with human drivers. Acting egoistically can make other drivers to take undesirable actions, such as performing hard brakes to avoid collisions. Designing a proper behavior involves dealing with antagonic objectives, such as increasing speed and avoiding rear-end collisions. Weighting those objectives in a reward or cost function is an error-prone and time-consuming task, and can become very hard as more features are added to the problem. This chapter presents an approach for designing autonomous vehicles behavior using learning from demonstration. A variation of the well-known Maximum Entropy Inverse Reinforcement Learning (IRL) algorithm is proposed in order to deal with continuous state spaces. Instead of exactly computing the gradients, we estimate them by sampling trajectories in regions with higher rewards using a Monte Carlo Tree Search (MCTS) based approach, which optimizes a Markov Decision Problem (MDP). We propose an interaction-aware MDP model capable of dealing with the inherent interaction and uncertainty present in surrounding vehicles motion. The experiments are performed in a merging scenario considering real data, showing that the proposed method can generate trajectories similar to the ones executed by human drivers. Additionally, favorable results are achieved when compared to traditional baseline methods and also to a variant of IRL that uses a polynomial-curve trajectory sampler.

Figure 30 – Maximum Entropy IRL via Monte Carlo Tree Search for autonomous driving in a merging scenario. The behavior of the ego-vehicle is learned from demonstrations by sampling trajectories according to the current reward function.



Source: Elaborated by the author.

5.1 Introduction

In many autonomous driving applications, the behavior of the autonomous vehicle (named *ego-vehicle* from now on for clarity) is determined by a reward or a cost function, which is a combination of *weights* and *features* (PADEN *et al.*, 2016). Nevertheless, manually tuning the weights can become a hard, tedious and error-prone task, because the ego-vehicle tries to optimize a function with antagonic objectives, such as keeping a target velocity and avoiding collisions with the vehicle in front. In this domain, Inverse Reinforcement Learning (IRL) (NG; RUSSELL *et al.*, 2000), which belongs to the class of *imitation learning* algorithms (HUSSEIN *et al.*, 2017), becomes a valuable tool since it allows learning optimal weights based on human drivers' expert demonstrations.

The demonstrations are assumed to be performed by an expert agent that solves a Markov Decision Process (MDP)(RUSSELL, 2010). Therefore, the objective of IRL is to recover the reward function that is being maximized by the expert agent. In general, the main IRL algorithms require solving the MDP problem to update the current weights at each learning iteration. This computation is necessary to compute the gradients by calculating the *features expectation*, which is a distribution over features given the current weights.

To overcome the computational burden of the MDP optimization, *trajectory sampling* can be applied to estimate the features expectation. These trajectories are rollouts of finite time episodes in which the initial features are encountered in the demonstrations dataset. Subsequently, the scene evolves according to a predefined policy executed by the ego-vehicle. Therefore, the samples are acquired by varying the policy, and changing the initial conditions according to the dataset so that the features expectation can be computed using the current weights. One key

aspect is that the chosen trajectory sampler must be capable of dealing with a highly dynamic and diverse environment as the one encountered in merging tasks with congested traffic. This task requires that the ego-vehicle be aware of surrounding vehicles' motion, while assuming some level level of uncertainty in their future states. Besides that, it is desirable that the sampler be efficient in order to diminish the number of sampled trajectories.

Addressing the main requirements needed to learn human drivers behaviors, the main contributions of this chapter can be summarized as follows:

- We apply Maximum Entropy Inverse Reinforcement Learning to learn a reward function in a merging scenario with congested traffic. The reward function is used to adapt the ego-vehicle policy to regions with higher rewards as in Fig. 30, increasing the learning efficiency. The results show that the optimal weights converge with a few number of sampled trajectories.

- To accomplish the aforementioned task, we propose an interaction-aware MDP model, which can estimate surrounding vehicles' future actions and account for uncertainty during this process. A policy is computed using a Monte Carlo Tree Search (MCTS) online solver (ŚWIECHOWSKI *et al.*, 2022). The proposed sampler is capable of planning near-optimal trajectories required to update the reward function.

- We compare the proposed method with polynomial curves-based trajectory planning and also with other baselines. The results indicate that the incorporation of complex reasoning in the sampler can achieve better learned behaviors.

The remaining of this chapter is organized as follows: previous works are discussed in Section 5.2 in order to highlight the contributions of this chapter; Section 5.3 describes the proposed methodology by presenting important concepts related to IRL, the sampling approach as well as the proposed interaction-aware MDP model; Section 5.4 details the experiments organization, implementation details and also the chosen features; in Section 5.5, we present quantitative and qualitative results related to our approach and baseline methods; finally, Section 5.6 highlights key aspects of this work and addresses future work as well.

5.2 Related Work

One of the most popular algorithms for IRL uses the principle of *maximum entropy* to handle the problem of sub-optimal demonstrations. Proposed by Ziebart *et al.* (2008), the approach considers that the demonstrations are sampled from a noisy distribution and the optimal weights are those which maximize the entropy of the system. The main drawback of this approach is the requirement of solving an MDP in the inner loop in order to calculate the feature expectation under the current rewards, making the algorithm very costly and even intractable for

large state spaces. Therefore, authors that make use of *Maximum Entropy* IRL in the automated driving domain have proposed some approaches to overcome this issue.

Kuderer, Gulati and Burgard (2015) employ IRL to fit a cost function of a trajectory planning that uses quintic polynomial splines. In order to make the problem tractable, they only compute the features of the most likely trajectory instead of computing the expectations given the current cost. Similar, González *et al.* (2018) compute trajectories using spatial-temporal lattices in order to achieve more complex behaviors while respecting the structure environment and kinematic constraints. However, by assuming only one optimal trajectory, these approaches lack the capability of capturing the diversity of trajectories generated by the same cost function.

Instead of approximating the feature expectation by the most likely trajectory, other authors chose to assume that the demonstrations are only local optimal by computing a second order Taylor approximation around the demonstrations (LEVINE; KOLTUN, 2012). Sadigh *et al.* (2016) apply this method to learn human drivers behavior to predict their motion during the interaction with the ego-vehicle. A similar approach is used by Schwarting *et al.* (2019). They model the surrounding vehicle behavior using Social Value Orientation (SVO), which quantifies the degree of an agent's selfishness or altruism, leading to a better prediction during interactions. Nevertheless, assuming only local optimal trajectories demonstrations is a strong assumption in a such large state space problem.

The drawback of the local optimality assumption can be handled by estimating the feature expectation from sampled trajectories (BOULARIAS; KOBER; PETERS, 2011). Rosbach *et al.* (2019) uses Model Predictive Control (MPC) to generate trajectories from sampling a set of actions, which approximates the samples to a uniform distribution. However, they do not consider other vehicles while learning the optimal weights, which limits its application to environments that requires interaction between vehicles. Wu *et al.* (2020) propose sampling trajectories by employing an efficient hierarchic sampler, since the computational burden is a concern in such approaches due to the number of samples required to estimate the expectations. It computes collision free paths via discrete elastic band, and subsequent calculates smooth paths and velocities to estimate the expectations. Nevertheless, they assume that surrounding vehicles follow a constant speed in the sampled trajectories. This drawback is addressed by Huang, Wu and Lv (2021), which apply the Intelligent Driving Model (IMD) (TREIBER; HENNECKE; HELBING, 2000) to model vehicles' behavior by explicitly accounting for interaction between them. The trajectories are sampled via a short-term planner using polynomial curves. Although considering interaction is closer to real scenarios, polynomial curves are sometimes a simplistic assumption for imitating complex behaviors. Moreover, the authors assume full knowledge about vehicles' motion that are not overridden by IDM, which are considered to follow their original trajectories in the dataset.

Additionally, one limitation of the aforementioned works is that they chose to sample trajectories independently of the current weights. This results in a large amount of samples to

estimate the features expectations. In this chapter, we propose to sample trajectories with respect to the current weights, moving the sample policy to regions of higher rewards, as in Finn, Levine and Abbeel (2016). They use a model-free optimization approach, which fits the dynamics of a manipulator robot using the expert and sampled trajectories, and posteriorly computes a time-varying linear-Gaussian controller. Nevertheless, their method only considers uncertainty in the control system and not in other agents behavior. In this regard, we propose an interaction-aware MDP model, which can capture the interaction between surrounding vehicles and also the uncertainty in their future motion. Moreover, we chose to re-planning the ego-vehicle's action at a given frequency in order to achieve more complex behaviors.

5.3 Methodology

5.3.1 Sample-based Maximum Entropy IRL

Inverse Reinforcement Learning (IRL) is the process of learning the reward function that describes an agent's behavior. The learning process requires a set of optimal trajectories $\mathcal{D}$ (demonstrations), which are provided by an expert. IRL considers that the expert is trying to maximize an underlined reward function. In this work, the reward function describes how the ego-vehicle behaves: higher rewards are related to those states where the ego-vehicle prefers to be.

Formally, a trajectory $\zeta = ((s_1,a_1),\ (s_2,a_2),\ ...)$ is a set of state-action pairs (s,a). The total reward of a trajectory is a combination of features $\mathbf{f}_\zeta = \sum_{s\in\zeta} \mathbf{f}_s$ and weight parameters θ, which maps the features into rewards:

$$R_\theta(\zeta) = \sum_s R_\theta(\mathbf{f}_s). \tag{5.1}$$

However, since considering that the demonstrations are optimal is a strong assumption, Ziebart et al. (ZIEBART *et al.*, 2008) consider that the expert demonstrations are sub-optimal trajectories sampled from a noisy distribution

$$P(\zeta) = \frac{1}{Z}\exp\left(R_\theta(\zeta)\right), \tag{5.2}$$

where $Z = \int \exp\left(R_\theta(\zeta)\right)d\zeta$ is the partition function and the optimal weights are those which maximize the entropy of the system:

$$\begin{aligned}
\theta^* &= \text{argmax}_\theta \mathcal{L}(\theta) \\
&= \text{argmax}_\theta \frac{1}{M}\sum_{\zeta\in\mathcal{D}} \log P(\zeta|\theta) \\
&= \text{argmax}_\theta \frac{1}{M}\sum_{\zeta\in\mathcal{D}} \log \frac{\exp\left(R_\theta(\zeta)\right)}{Z} \\
&= \text{argmax}_\theta \frac{1}{M}\sum_{\zeta\in\mathcal{D}} R_\theta(\zeta) - \log Z,
\end{aligned} \tag{5.3}$$

where M is the number of expert trajectories.

In small and discrete state spaces, the partition function can be exactly computed by using a dynamic programming algorithm, avoiding the need of sampling many possible trajectories to estimate Z (ZIEBART *et al.*, 2008). However, this computation can become very costly and even intractable for large state spaces as it requires solving an MDP in the inner loop, since θ is iteratively updated given the gradients.

Other authors propose to estimate Z instead of computing it exactly (BOULARIAS; KOBER; PETERS, 2011)(KALAKRISHNAN *et al.*, 2013)(FINN; LEVINE; ABBEEL, 2016). If the set of sampled trajectories $\tilde{\mathcal{D}}$ is assumed to be finite and sampled from a chosen background distribution $q(\zeta)$, the gradients can be given by

$$\nabla_\theta \mathcal{L}(\theta) = \frac{1}{M} \sum_{\zeta \in \mathcal{D}} \mathbf{f}_\zeta - \frac{1}{Z} \sum_{\zeta \in \tilde{\mathcal{D}}} w_\zeta \mathbf{f}_\zeta, \tag{5.4}$$

where $w_\zeta = z(\zeta) \exp(R_\theta(\zeta))$, $Z = \sum_\zeta w_\zeta$, $z(\zeta)$ are the importance weights related to the distribution $q(\zeta) \propto \exp(R_\theta(\zeta))$, and $R_\theta(\mathbf{f}_\zeta)$ is considered linear with respect to θ. With the gradients, θ can be iteratively updated using gradient descent.

Specifically in datasets containing human drivers trajectories, each demonstration represent a particular scenario with different initial conditions. Since the demonstrations have equal probability of being observed, the gradient must be slightly modified as:

$$\nabla_\theta \mathcal{L}(\theta) = \frac{1}{M} \sum_{\zeta \in \mathcal{D}} \left[\mathbf{f}_\zeta - \frac{1}{Z} \sum_{\tilde{\zeta}_i=0}^{|\tilde{\mathcal{D}}|} w_{\tilde{\zeta}_i} \mathbf{f}_{\tilde{\zeta}_i} \right] - 2\lambda\theta, \tag{5.5}$$

where $\tilde{\zeta}_i$ are sampled trajectories with the same initial conditions as ζ, and $|\tilde{\mathcal{D}}|$ is the amount of samples. We also add a L2 normalization term $-2\lambda\theta$ in the gradient to prevent overfitting, where λ is the L2 normalization weight. In this work, the trajectories are generated based on the current value of θ. This attempts to generate more samples with higher rewards according to the current R_θ. Since trajectories are sampled from multiple distributions, the importance weights can be estimated by computing a fusion distribution (FINN; LEVINE; ABBEEL, 2016),

$$z(\tilde{\zeta}_i) = \frac{1}{J} \sum_{j=0}^{J} \frac{1}{q_j(\tilde{\zeta}_i)}, \tag{5.6}$$

where J is the number of distributions. In this work, $J = |\tilde{\mathcal{D}}|$ as $q(\tilde{\zeta}_i) \propto \exp(R_\theta(\tilde{\zeta}_i))$ only varies with respect to the current weights, which are updated at each iteration of the algorithm. Some authors chose $q(\zeta)$ to be unique, such as Rosbach et al. (ROSBACH *et al.*, 2019), Wu et al. (WU *et al.*, 2020) and Huang et. al (HUANG; WU; LV, 2021), in which all trajectories are sampled from the same distribution before the training initiates. Instead, we propose to use an adaptive sampling: the IRL algorithm is interleaved with an online policy optimization step using MCTS, which generates more samples with higher rewards according to the current θ.

5.3.2 MCTS Sampler

During navigation, the ego-vehicle must adopt some tactical behaviors in order to safely reach its destination. This task becomes more complex when other traffic participants are considered. For example, merging onto an adjacent lane requires a higher level of reasoning than deviating from a static obstacle on the road. Thus, the ego-vehicle must consider its future actions as well as other vehicles' motion. This prediction comes along with uncertainty, assuming that communication between vehicles is not available. In order to deal with the inherent uncertainty, the problem can be modeled as a Markov Decision Process (MDP) (BRECHTEL; GINDELE; DILLMANN, 2011).

However, the optimal policy computation in MDPs becomes very costly and even intractable for models with large state spaces by using traditional methods, such as value iteration and policy iteration (RUSSELL, 2010). Thus, an approach that can handle large state spaces while ensuring, at least, near-optimal policies must be employed. In this work, an online solver based on Monte Carlo Tree Search (MCTS) (ŚWIECHOWSKI *et al.*, 2022) is used to compute near-optimal policies in MDPs, similar to POMCP solver detailed in Chapter 2, but without assuming non-observable states.

In the remaining of this section, the key components of the MDP model proposed in this chapter is described.

5.3.2.1 State Space

The state space $\mathcal{S}$ includes the states of all vehicles considered in the scene,

$$\mathcal{S} = [s_0 \quad s_1 \quad s_2 \quad \cdots \quad s_K]^{\mathrm{T}}, \tag{5.7}$$

where s_0 represents the state of the ego-vehicle and $s_k \in \{1, \cdots, K\}$ are the states of surrounding vehicles. A vehicle is considered in the scene when its distance to the ego-vehicle is within 30 m. Therefore, the number of surrounding vehicles can vary at each time step.

The vehicle's state $\mathcal{S}_k$ is defined as

$$s_k = [d_k \quad u_k \quad v_{d\_k}]^{\mathrm{T}}, \tag{5.8}$$

where d_k is the longitudinal distance from the start of the road to the vehicle center; u_k is the lateral distance from the center of the right-most lane to the vehicle center; and v_{d_k} is the longitudinal speed.

5.3.2.2 Action Space and Transition Model

The transition on the state variables related to the vehicles' lateral and longitudinal motion is constrained to the following motion model:

$$\begin{bmatrix} d'_k \\ v'_{d\_k} \\ u'_k \end{bmatrix} = \begin{bmatrix} d_k \\ v_{d\_k} \\ u_k \end{bmatrix} + \begin{bmatrix} v\_d_k \\ a_{d\_k} \\ v_{u\_k} \end{bmatrix} \Delta t + \frac{1}{2} \begin{bmatrix} a_{d\_k} \\ 0 \\ 0 \end{bmatrix} \Delta t^2, \tag{5.9}$$

in which a_{d_k} and v_{u_k} is the speed rate change and lateral speed of vehicle k, respectively, and Δt is the time step. The ego-vehicle's longitudinal acceleration a_{d_0} is chosen based on UCT criterion, where a_{d_0} can be selected from $\mathcal{A}_{a_{d0}} = \{-1.0,\ -0.5,\ 0,\ 0.5,\ 1.0\}\text{m/s}^2$. On the other hand, surrounding vehicles are assumed to plan their speed according to the IDM model (TREIBER; HENNECKE; HELBING, 2000):

$$a_{IDM} = a_{max}\left(1 - \left(\frac{v_k}{v_{desired}}\right)^{\delta} - \left(\frac{g^*(v,\ \Delta v)}{g}\right)^2\right) \tag{5.10}$$

$$g^*(v,\ \Delta v) = g_{desired} + t_{tc}v + \frac{v\Delta v}{2\sqrt{a_{max}b_{comfort}}}, \tag{5.11}$$

where the rear-to-front distance g and the speed difference Δv are computed with respect to either the vehicle in front or end of its current lane. Also, δ is the acceleration exponent, a_{max} is the desired maximum acceleration, $b_{comfort}$ is the comfort deceleration, $v_{desired}$ is the desired speed and t_{tc} is the desired time gap. In order to account for model uncertainties and physical constraints,

$$a_{d\_k} = \max\left(a_{IDM} + \mathcal{N}(0, \sigma^2_{acc}),\ -b_{safe}\right) \tag{5.12}$$

where $\mathcal{N}(0, \sigma^2_{acc})$ is a zero mean Gaussian noise acceleration and b_{safe} is the maximum safe deceleration.

The lateral position of surrounding vehicles is considered constant by making $\{v_{u\_k} = 0 \mid \forall k \neq 0\}$. As in (HUBMANN *et al.*, 2018b), we consider that the ego-vehicle can instantaneously change its lateral speed v_{u_0} according to $\mathcal{A}_{0\_lat} = \{LC,\ SCL\}$, where:

$$\begin{cases} v_{u\_0} = & \min(0.17v_{d\_k},\ 0.6), & \text{if } a_{0\_lat} \text{ is LC} \\ v_{u\_0} = & 0, & \text{otherwise} \end{cases}, \tag{5.13}$$

in which LC and SCL represent a change to the target lane, and staying in the current lane, respectively.

5.3.2.3 Reward Model

As discussed in Section 5.3.1, the reward model is a linear combination of features $\mathbf{f}_s$ and a set of learned weights θ. The features must encode important aspects of human driver behaviors so that the reward model be capable of imitating their trajectories. The set of chosen features are detailed in Section 5.4.

5.3.3 Algorithm Summary

The summary of the proposed Maximum Entropy IRL via MCTS trajectory sampling algorithm (MEIRL-MCTS) can be seen in Algorithm 1.

5.4 Experiment Setup

5.4.1 Dataset

The training data is selected from the subset DR_CHN_Merging_ZS0 belonging to the INTERACTION dataset (ZHAN *et al.*, 2019), as shown in Fig. 31. The dataset provides the position in global coordinates, the heading and the speed of the vehicles in the scene. This information is used to reconstruct the road structure throughout a whole trajectory. The duration of each trajectory is 4s duration and the sampling time $\Delta t_{sim} = 0.1$s. The chosen merging

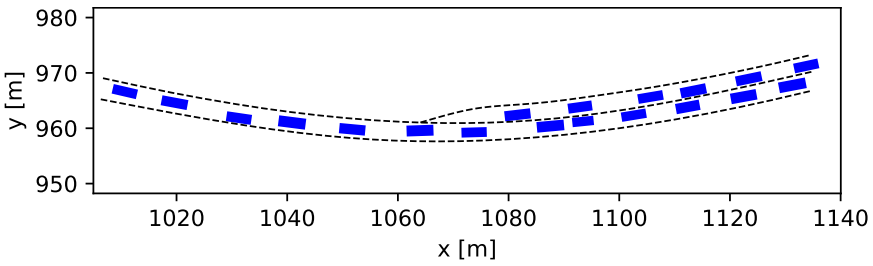
Algorithm 1: Maximum Entropy Inverse Reinforcement Learning via MCTS trajectory sampling

Input : Human drivers demonstrations $\mathcal{D}$, environment model $\mathcal{C}$, interaction-aware MDP model, MCTS online solver, learning rate α, regularization parameter λ, number of iterations I and number of epochs E.

Result: optimized reward weights θ^*

Initialize $\theta \leftarrow$ *random*([-1, 1]);
Compute feature expectation for expert demonstrations $\bar{\mathbf{f}}(\mathcal{D}) = \frac{1}{M}\sum_{\zeta\in\mathcal{D}}\mathbf{f}_\zeta$;
Initialize the samples datasets $\tilde{\mathcal{D}}_{i=0:M} \leftarrow [\,]$;
Initialize the reward weights buffer $\Theta \leftarrow [\,]$;
for *iteration* $\leftarrow 1$ *to* I **do**
    foreach $\zeta_i \in \mathcal{D}$ **do**
        Generate a trajectory $\tilde{\zeta}_i$ with the same initial conditions as ζ_i using the MDP model, the MCTS online solver and environment model $\mathcal{C}$;
        Add $\tilde{\zeta}_i \rightarrow \tilde{\mathcal{D}}_i$;
    end
    Add $\theta \rightarrow \Theta$;
    For each initial condition, calculate $q_j(\tilde{\zeta}_i) = \frac{\exp(R_{\theta_j}(\tilde{\zeta}_i))}{\sum_{i=0}^{|\Theta|}\exp(R_{\theta_j}(\tilde{\zeta}_i))}$ for each $\theta_j \in \Theta$;
    Compute the importance weights $z(\tilde{\zeta}_i) = \frac{1}{|\Theta|}\sum_{j=0}^{|\Theta|}\frac{1}{q_j(\tilde{\zeta}_i)}$;
    for *epoch* $\leftarrow 1$ *to* E **do**
        Compute the features expectation with the sampled trajectories for each initial condition: $\tilde{\mathbf{f}} = \frac{1}{Z}\sum_{\tilde{\zeta}_{i=0}}^{|\tilde{\mathcal{D}}_i|} w_{\tilde{\zeta}_i}\mathbf{f}_{\tilde{\zeta}_i}$;
        Calculate the gradient $\nabla_\theta\mathcal{L}(\theta) = \frac{1}{M}\sum_{i=0}^{M}\left[\bar{\mathbf{f}}_i(\mathcal{D}) - \tilde{\mathbf{f}}_i(\tilde{\mathcal{D}})\right] - 2\lambda\theta$;
        Update the reward weights $\theta \leftarrow \theta + \alpha\nabla_\theta\mathcal{L}(\theta)$;
    end
end

Figure 31 – Subset DR_CHN_Merging_ZS0 in the INTERACTION dataset. The vehicles are moving from right to left.



Source: Elaborated by the author.

scenario is composed of two lanes: the length of the right lane and the left lane are 75m and 155m, respectively, and the width of both lanes is 3.4m. This means that the vehicles on the right must change lane to continue its travel. Finally, 100 trajectories are selected as training data and 40 trajectories as test data. The trajectories are selected from different locations in the scene to provide a diverse training and test data.

5.4.2 Implementation Details

The MCTS and the MDP model are implemented using *POMDP.jl* (EGOROV *et al.*, 2017), a framework for sequential decision making under uncertainty. *POMDP.jl* is implemented in *Julia*, a high-level, dynamic and high-performance language. On the other hand, the environment model required to simulate the sampled trajectories, and also the Maximum Entropy IRL algorithm are implemented using *Python*. During the simulation, vehicles on the same lane and behind the ego-vehicle are overridden by IDM to simulate a more realistic interaction behavior. The values of the IDM parameters are $g_{desired} = 0.8$m, $t_{tc} = 0.6$s, $a_{max} = 0.8$m/s^2, $b_{comfort} = 1.8$m/s^2, $v_{desired} = 7.0$m/s and $b_{safe} = 4$m/s^2. The other vehicles follow their original trajectories in the dataset. The system containing the simulation environment and algorithms runs on a Intel Core i7-6500U CPU CPU with 2.50 Hz, a 7.7 GB memory RAM and a NVIDIA GeForce 930M. The code is available on https://bitbucket.org/juniorars/pomdp_irl/src/master/.

For simplification, the MDP model only considers vehicles that are 30m ahead, and the the vehicles behind it on the same and target lanes, respectively. The main MDP model and MCTS parameters are $c = 1.0$, $\gamma = 0.99$, $\Delta t = 1.0$s and $\sigma^2_{acc} = 0.05$m/s^2. The IDM parameters used in the MDP model are the same as of the simulation. Additionally, we need to smooth the computed trajectory, since the simulation runs at a higher frequency than the MCTS solver. Therefore, we employ a ramp-based trajectory planning for longitudinal acceleration and lateral speed according to the their current values as well as the target values computed by the MCTS solver. The proposed algorithm runs for 10 iterations, where the reward weights are trained using 1e4 epochs with a L2 normalization parameter $\lambda = 0.001$ and learning rate $\alpha = 0.05$.

5.4.3 Feature Selection

As discussed in Section **??**, the reward model is a linear combination of $\mathbf{f}_s$ and θ. Therefore, the features must be selected in such a way to describe the behavior of the human drivers and capture important aspects of their driving styles. The features considered in this work are detailed below:

5.4.3.1 Speed

This feature is directly related to the ego-vehicle progress along the road:

$$f_v(s_t) = (v(t) - v_{ref})^2, \tag{5.14}$$

where $v_{ref} = 7$m/s is the reference speed of the ego-vehicle.

5.4.3.2 Comfort

The comfort level is directly associated to longitudinal acceleration and lateral speed. Therefore,

$$f_j(s_t) = \left(\frac{a_s(t) - a_s(t-1)}{\Delta t_{sim}}\right)^2, \tag{5.15}$$

$$f_l(s_t) = \left(\frac{v_u(t) - v_u(t-1)}{\Delta t_{sim}}\right)^2. \tag{5.16}$$

5.4.3.3 Merging

The merging feature is proposed to represent the necessity of changing lane. As the ego-vehicle approaches the end of the right lane, it must move towards the left lane to avoid stopping. Assuming that, the merging feature is given by:

$$f_m(s_t) = \left(1 - \frac{min(u(t),\, W)}{W}\right).\left(\frac{1 - max(0,\, L - d(t))}{L}\right)^2, \tag{5.17}$$

where L is the length of the right-most lane and W is width of the lanes. The first and second factors are related to the lateral and longitudinal progress, respectively. The idea behind this feature is that it does not have any influence when the ego-vehicle reaches the left lane and low influence when it is far from the end of the right lane.

5.4.3.4 Time gap

Human drivers tend to slow down to avoid staying too close to other vehicles. By considering this feature, the ego-vehicle can be aware of potential future read-end collisions and adapt the speed according to the distance to the vehicle in front (HUANG; WU; LV, 2021):

$$f_g(s_t) = \exp\left(-\frac{d_{ego}(t) - d_{front}(t)}{v_{ego}(t)}\right). \tag{5.18}$$

5.4.3.5 Interaction

The interaction with other vehicles is an important factor to consider. The ego-vehicle must avoid forcing other vehicles to perform hard deceleration, since it can cause some level of discomfort to them:

$$f_i(s_t) = (a_{back})^2, \text{if } a_{back} < 0, \tag{5.19}$$

where a_{back} is the longitudinal acceleration of the vehicle that is following the ego-vehicle.

5.4.3.6 Collision

Although $f_g(s_t)$ and $f_i(st)$ can balance the distances between following and leader vehicles, this feature is import to avoid sideswipe accidents. The collision feature is defined as:

$$f_c(s_t) = \begin{cases} 1, & \text{if collision,} \\ 0, & \text{otherwise.} \end{cases} \tag{5.20}$$

Since the features have different measure units and scale, they are all normalized to $[0,1]$ based on the minimum and maximum values of each feature in the demonstrations.

5.4.4 Baseline methods

To validate our method, we compare the MCTS approach with different models that are either used to sample trajectories to estimate the feature expectation or are directly applied on the ego-vehicle's control:

5.4.4.1 Polynomial curve-based sampling

This sampler is based on state space-based planning, in which the trajectories are computed according to predefined goal states in a short time horizon (WERLING *et al.*, 2010; FERGUSON; HOWARD; LIKHACHEV, 2008). The goal states generally considers the end lateral position on the road, and the end speed. To ensure smooth transition between states for each trajectories, state space-based planner often use polynomial, bézier, splines or spirals curves (GONZÁLEZ *et al.*, 2015). For comparison, we apply the polynomial curve-based sampler proposed in (HUANG; WU; LV, 2021), in which the trajectories are represented by two polynomial functions with respect to x and y coordinates:

$$\begin{cases} d = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4, \\ u = b_0 + b_1 t + b_2 t^2 + b_3 t^3 + b_4 t^4 + b_5 t^5, \end{cases} \tag{5.21}$$

where d and u are the longitudinal and lateral positions, respectively. The degree of the polynomial functions allows the control over the end longitudinal speed and acceleration, as well as over the end lateral position, speed and acceleration. The coefficients of each polynomial can be determined by solving a linear system based on its derivatives, and initial and goal

states. The reader is refereed to (HUANG; WU; LV, 2021) for more details. The initial states are the same of the demonstrations, whereas the goal states are the end longitudinal speed and acceleration v_{de}, a_{de}, and the end lateral position u_e. The remaining goal states are set to zero. We found necessary not to set a_{de} to zero in the goal states so that to better adapt the planner to the demonstrations. Therefore, the sampling set of goal states are $v_{de} = [v_{ds} - 4,\ v_{ds} + 4]$ with a interval of $1m/s$, $a_{de} = \{a_{ds} - 1,\ a_{ds},\ a_{ds} + 1\}$ and $u_e = [u_s,\ u_w]$ with a interval of $\frac{u_w - u_s}{5}$m, where $u_w = W$.

5.4.4.2 IDM+MOBIL

Whereas IDM is widely applied to following vehicle behavior, Minimizing Overall Braking Induced by Lane Change (MOBIL) (KESTING; TREIBER; HELBING, 2007) is useful to describe whether a lane change is desirable or not. MOBIL considers the advantage in performing a lane change and its impact on the new back vehicle (the vehicle behind the ego-vehicle on the target lane):

$$\begin{aligned} a'_{d\_ego} - a_{d\_ego} &> p(a'_{d\_back} - a_{d\_back}) + a_{thr}, \\ a'_{d\_back} &> -b_{safe}, \end{aligned} \tag{5.22}$$

where a_d and a'_d are longitudinal accelerations before and after a possible lane change, respectively. The politeness factor $p = [0, 1]$ controls the ego-vehicle aggressiveness by decreasing the back vehicle disadvantage. The threshold acceleration a_{thr} is added to avoid lane changes when the advantage is not significant. For the experiments, we adopt $p = 0.1$ and $a_{thr} = 0.1$, where the IDM parameters are the same as described in Section 6.4.2.

5.4.4.3 Constant speed

This baseline simply assumes that the ego-vehicle follows a constant speed and stay on the current lane for the whole trajectory.

5.4.5 Evaluation Metrics

The main metrics used to compare the different approaches are feature deviation and mean Euclidean distance.

5.4.5.1 Feature deviation

The goal of IRL is to imitate expert demonstrations by computing a feature distribution similar of the one in the dataset. The feature deviation (FD) can be computed as

$$\mathcal{E}_{FD} = \frac{1}{M} \sum_{i=1}^{M} \frac{1}{N_i} |\mathbf{f}(\zeta_i^{gt}) - \mathbf{f}(\zeta_i^{plan})|, \tag{5.23}$$

where M is the number of expert demonstrations and N_i is the length of the i-th trajectory.

5.4.5.2 Euclidean Distance (ED) and Mean Euclidean Distance (MED)

The Euclidean Distance (ED) is computed by considering the final position of the ego-vehicle for a trajectory in the demonstration and the trajectory given by the sampler with the same initial conditions:

$$\mathcal{E}_{ED} = ||\zeta_i(N_i)^{gt} - \zeta_i(N_i)^{plan}||_2. \tag{5.24}$$

Similarly, the Mean Euclidean Distance (MED) is given by:

$$\mathcal{E}_{MED} = \frac{1}{M}\sum_{i=1}^{M} ||\zeta_i(N_i)^{gt} - \zeta_i(N_i)^{plan}||_2. \tag{5.25}$$

5.4.5.3 Qualitative analysis

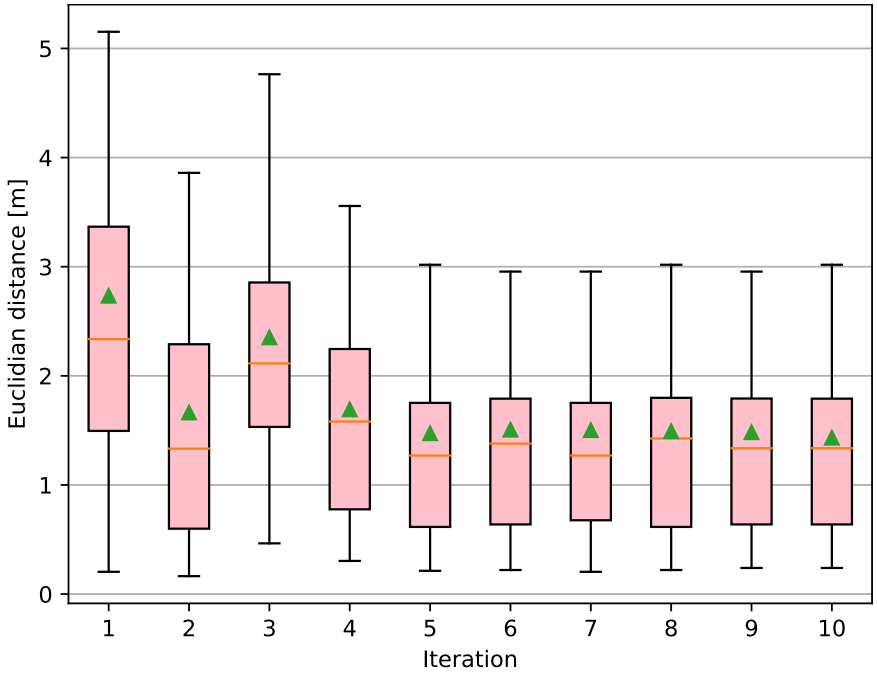
We also analyze the quality of the trajectories using the learned reward weights. This aims to verify the similarity of the planned trajectories with the demonstrations, as well as to show the smoothness of the control inputs applied to the ego-vehicle.

5.5 Results

5.5.1 Learning analysis

We evaluate the evolution of the learning rewards with respect the iterations according to the proposed metrics. As discussed in Section 5.3.1, the trajectories are sampled to estimate the

Figure 32 – ED distributions with respect the MEIRL-MCTS iterations. The algorithm is capable of decreasing the ED as more trajectories are sampled.

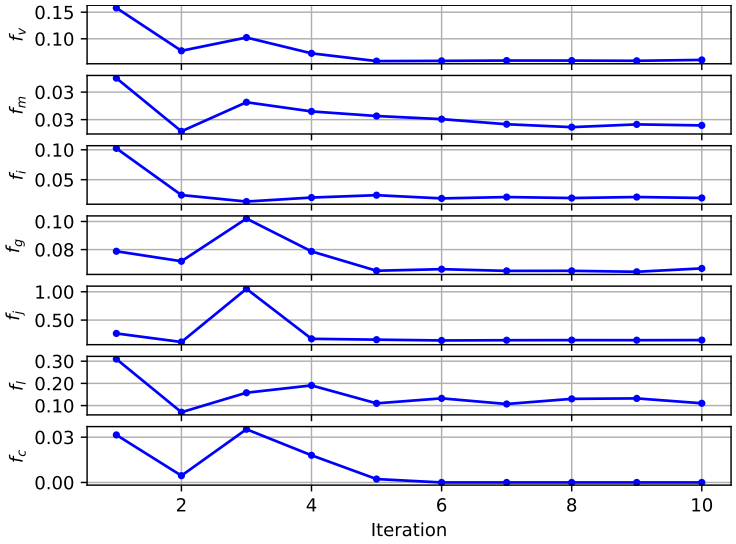


Source: Elaborated by the author.

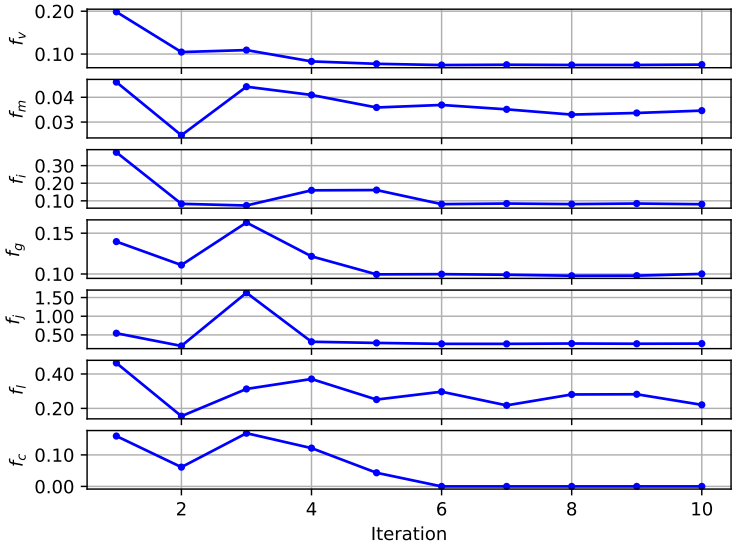
feature expectation according to the current rewards in order to minimize the error with respect to the demonstrations.

Fig. 32 shows the ED for each iteration represented by box-plots, which helps to better visualize the ED distribution. As we can see, the ED oscillates at the beginning of the learning and converges after a few iterations. This occurs because, at each iteration, the new sampled trajectories bounds the feature error by adapting the feature expectation according to the learned rewards. This statement is corroborated by the analysis of the feature error mean and standard deviation, as depicted in Fig. 33. We can see that the overall error decreases as more trajectories

Figure 33 – Evolution of FD with respect the MEIRL-MCTS iterations.



(a) Mean.



(b) Standard deviation.

Source: Elaborated by the author.

are added to the samples. However, the error of one particular feature may increase between the iterations. It is expected since some features have antagonic effects, meaning that the adjustment of one feature weight can influence all features expectation.

5.5.2 Performance on the Test Set

5.5.2.1 Feature Deviation

Table 11 details the FD for the four methods. i.e. constant speed behavior, IDM+MOBIL, polynomial curve planner and the proposed MEIRL-MCTS. Both planners utilize their learning rewards in the test trajectories. However, the state-space planner assumes full knowledge about the motion of surrounding vehicles during the planning horizon and thus does not make use of any prediction model. It computes the total reward for each sampled trajectory and chose the one with maximum reward. It is important to mention that this process does not occur in real-time. As we can see, our method presents relatively smaller FD when compared with the baselines, with the exception of f_i and f_l. These results were already expected for both constant speed behavior and IDM+MOBIL since they are too simple to deal with a such complex scenario. However, our method also presented a better performance with respect to the state-space planner, even with its full knowledge about the surrounding vehicles' future positions. One possible reason is the efficiency of the state-space planner. We noted that many sample trajectories result in collisions or are kinematic infeasible. As a consequence, the samples cannot make a good estimation of the feature expectation.

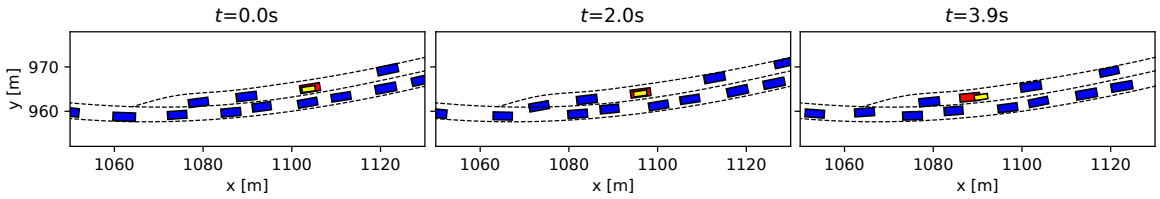
5.5.2.2 Mean Euclidean Error

The analysis of the MED is similar to the FD, with our method being consistently better than the baselines. This shows that there is a correlation between FD and MED, which demonstrates that the chosen features can well describe human driver behaviors during merging.

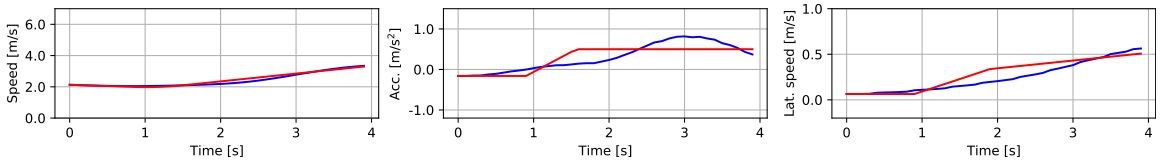
Table 11 – Results in the test set.

	f_v	f_m	f_i	f_g	f_j	f_l	f_c	MED
MEIRL-MCTS	0.058± 0.72	0.023± 0.032	0.028± 0.189	0.052± 0.083	0.097± 0.157	0.105± 0.272	0.000± 0.000	1.357± 0.906
Polynomial	0.075± 0.095	0.029± 0.038	0.020± 0.085	0.053± 0.085	0.100± 0.145	0.082± 0.236	0.000± 0.000	1.734± 1.584
IDM+MOBIL	0.126± 0.142	0.052± 0.059	0.075± 0.412	0.077± 0.109	52.01± 203.42	2.378± 16.17	0.000± 0.000	3.069± 3.400
Const. Speed	0.086± 0.106	0.056± 0.086	0.052± 0.355	0.083± 0.147	0.065± 0.110	0.357± 3.124	0.009± 0.096	2.077± 1.726

Figure 34 – Example of trajectory in which the ego-vehicle is attempting to merge onto the target lane. However, it cannot find a proper gap to conclude the maneuver.



(a) Trajectory positions. The controlled ego-vehicle is colored in red and the demonstration is in yellow. Surrounding vehicles are in blue.



(b) Controlled ego-vehicle states with respect to time. The controlled ego-vehicle is colored in red and the demonstration is in blue.

Source: Elaborated by the author.

5.5.3 MCTS Trajectory analysis

In this section, we demonstrate the efficiency of the proposed MEIRL-MCTS by analyzing trajectories planned according to the learned rewards. This analysis takes into account the ego-vehicle's longitudinal speed and acceleration, as well as its lateral speed. The objective is to show the smoothness of the planned trajectories and also compare them with the demonstrations. Figures 34, 35 and 36 depict three representative trajectories from the MEIRL-MCTS and the demonstrations as well.

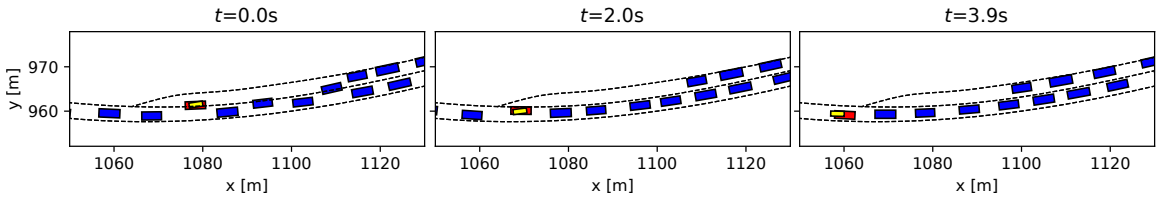
5.5.3.1 Attempt to merge

Fig. 34 shows the ego-vehicle approaching the end of the left lane and moving towards the right lane in the attempt to merge. This move occurs because the ego-vehicle perceives that a possible gap is forming on the target lane. As we can see, the planned trajectory is similar to the demonstration, with a small final position displacement. Also, the controls applied to the ego-vehicle are continuous and absent of abrupt variations.

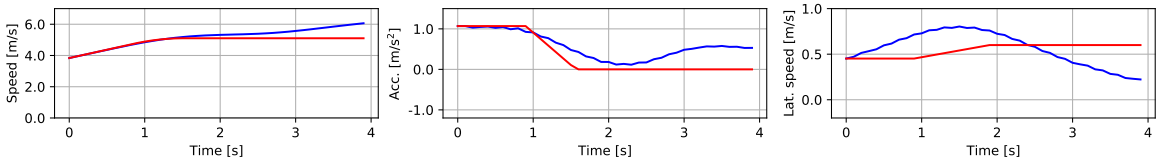
5.5.3.2 Merging

In Fig. 35, the ego-vehicle performs a lane change and stops accelerating to avoid staying too close to the vehicle in front. This example demonstrates that the MEIRL-MCTS can perform safe merging maneuvers as the one observed in the demonstration.

Figure 35 – Example of a trajectory in which the ego-vehicle merges onto the target lane.



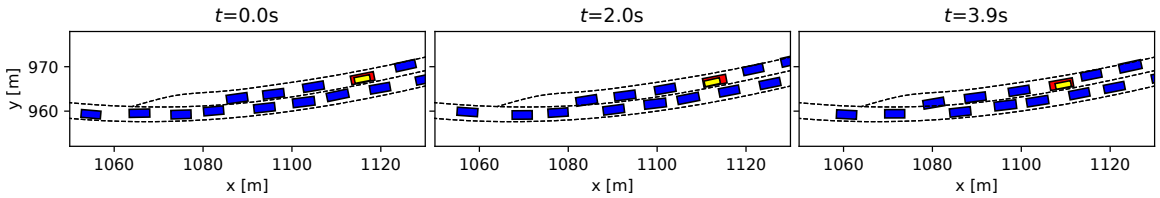
(a) Trajectory positions. The controlled ego-vehicle is colored in red and the demonstration is in yellow. Surrounding vehicles are in blue.



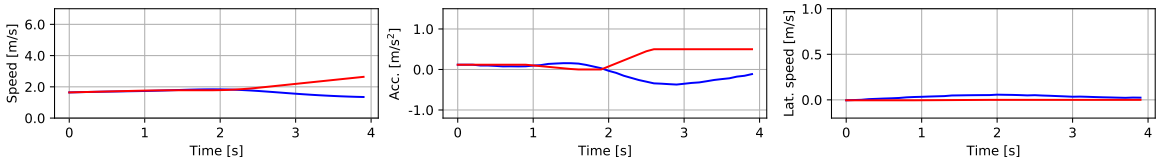
(b) Controlled ego-vehicle states with respect to time. The controlled ego-vehicle is colored in red and the demonstration is in blue.

Source: Elaborated by the author.

Figure 36 – Example of trajectory of leader following behavior. Since the ego-vehicle is far from the end of lane, it does not force a lane change maneuver.



(a) Trajectory positions. The controlled ego-vehicle is colored in red and the demonstration is in yellow. Surrounding vehicles are in blue.



(b) Controlled ego-vehicle states with respect to time. The controlled ego-vehicle is colored in red and the demonstration is in blue.

Source: Elaborated by the author.

5.5.3.3 *Leader following*

In particular situations, a lane change is not desirable, since it can be dangerous for the ego and surrounding vehicles, as in Fig. 36. This is often observed when the ego-vehicle is too far from the end of the lane. In this case, the vehicles on the adjacent lane does not tend to act cooperatively and does not give enough room for a lane change. Moreover, the values of

f_m, which is mainly responsible for the merging, are very low in this situation. Therefore, the ego-vehicle can stay on its current lane and avoid dangerous maneuvers.

5.6 Final Considerations

This chapter presents a variation of the Maximum Entropy IRL algorithm that uses an MCTS online solver to sample trajectories in order to estimate the gradient of the IRL algorithm. The MCTS solves an MDP problem that considers interaction between vehicles, resulting in a better estimation of surrounding vehicles' position. The main objective of the MCTS planner is to sample trajectories according to the current rewards. The proposed approach is used to learning human driving behaviors in a merging scenario. The results show better performance compared to baseline methods and also that the planned trajectories can be applied to real autonomous systems.

Currently, the proposed approach assumes that the MDP model have perfect knowledge of surrounding vehicles' behavior, by considering that their actions are planned according to IDM. In some occasions, this can lead to poor vehicles' prediction. For example, the vehicle on the target lane behind the ego-vehicle may act cooperatively and decelerate to give room for a merging. In the next chapter, an extension of the interaction model that consider such behavior is considered.

CHAPTER

6

LEARNING DRIVING BEHAVIOR FOR AUTONOMOUS VEHICLES IN PARTIALLY OBSERVABLE ENVIRONMENTS

Artificial Intelligence (AI) is crucial in the field of autonomous driving because it enables vehicles to perceive, reason, and act in complex driving scenarios. AI algorithms can analyze and reason about the data from sensors to make decisions that maximize safety and efficiency, such as when to change lanes, brake, accelerate, or make a turn. Moreover, AI can also be used to make autonomous vehicles behave in a similar way as human drivers. This is crucial to achieve a smooth joint navigation with other vehicles, since the task of driving relies on social rules inherent to human driving behavior. This chapter presents an approach for designing autonomous vehicles behavior using learning from demonstration. A variation of the well-known Maximum Entropy Inverse Reinforcement Learning (IRL) algorithm is proposed in order to deal with continuous and partially observable state spaces. Instead of exaclty computing the gradients, we estimate them by sampling trajectories in regions with higher rewards using a Partially Observable Markov Decision Process (POMDP) based approach. We propose an interaction-aware POMDP model capable of estimating surrounding vehicles intention during the planning in order to achieve better sample policies according to the current rewards. Results show that the proposed approach compares favorably to deterministic methods, in which surrounding vehicles latent intentions are not considered in the IRL problem formulation.

6.1 Introduction

Inverse Reinforcement Learning (IRL) is an area of machine learning that enables machines to learn behavior by observing and imitating the actions of a human expert. It is based on the idea that humans are capable of making decisions in complex situations, so it can be used to teach robots or other artificial agents how to make similar decisions. In IRL, a reward

function is learned from observed human behavior and then used as feedback for an agent's learning process. This allows IRL-based systems to develop strategies and behaviors that reflect those of their teachers without being explicitly programmed with them. Those features make IRL extremely useful in autonomous driving domain, since it allows to mimic human driving styles.

The main IRL algorithms require solving a Markov Decision Process (MDP)(RUSSELL, 2010) problem to update the current reward weights at each learning iteration, since the expert trajectories are considered MDP optimal policies. By making use of MDP, this formulation assumes that the state space is fully observed by the agent. Nevertheless, in highly interactive driving scenarios, human drivers need to infer *the intention* of each other by making sequential observations of the environment, which is crucial to perform safe and efficient trajectories. In this sense, if we consider that human drivers intentions are only partially observable, the expert agent is no longer optimizing an MDP problem. Instead, it is solving a Partially Observable Markov Decision Process (POMDP), which is a mathematical model for dealing with partially observable environments (KAELBLING; LITTMAN; CASSANDRA, 1998). Hence, the main hypothesis presented in this chapter is that modeling the IRL problem as a POMDP can improve the learning performance when compared with traditional MDP approaches.

This work can be seen as an extension of our previous work (SILVA; GRASSI; WOLF, 2023). The new contributions presented in this chapter can be summarized as follows:

- In order to solve the forward partially observable problem in the IRL loop, we propose an interaction-aware POMDP model that is capable of infer whether surrounding vehicles are giving the right of way to the ego-vehicle. The results show that our model can efficiently predict the intention of human drivers.

- Maximum Entropy Inverse Reinforcement Learning is applied to learn a reward function in a highly interactive merging scenario. The reward function is used to adapt the computed policy to regions with higher rewards, increasing the learning efficiency. The POMDP model is used in the online solver Partially Observable Monte Carlo Planning (POMCP) (SILVER; VENESS, 2010) to sample near-optimal policies in real-time according to the current rewards. Quantitative results show that our method outperforms approaches in which the intention of other vehicles are considered fully observable.

- To deal with the high computation complexity in POMDPs, we present a simple constant action heuristic to evaluate the value of reached beliefs, instead of using traditional random rollout policies. The experiments show that the adoption of this heuristic can highly impact the learning efficiency.

The remaining of this chapter is organized as follows: previous work are discussed in Section 6.2 in order to highlight the contributions of this chapter; Section 6.3 describes the work methodology by presenting important concepts related to IRL, the sampling approach as well as

the proposed POMDP model; Section 6.4 details the experiments organization, implementation details and also the chosen features; in Section 6.5, we present quantitative and qualitative results related to our approach and baseline methods; finally, Section 6.6 highlights key aspects of this work and addresses future work as well.

6.2 Related Work

This section discusses works that apply maximum entropy IRL (ZIEBART *et al.*, 2008) to autonomous driving, and also how they approach the drawback of solving an MDP in the inner loop of the algorithm in order to calculate the feature expectation. Moreover, works that propose solving the IRL problem by considering that the dynamics are modeled as a POMDP are considered.

6.2.1 IRL in autonomated driving domain

Kuderer, Gulati and Burgard (2015) employ IRL to fit a cost function of a trajectory planning that uses quintic polynomial splines. In order to make the problem tractable, they only compute the features of the most likely trajectory instead of computing the expectations given the current cost. This simplification can be overcome by estimating the feature expectation by trajectory sampling. Wu *et al.* (2020) propose sampling trajectories by employing an efficient hierarchic sampler, since the computational burden is a concern in such approaches due to the number of samples required to estimate the expectations. It computes collision free paths via discrete elastic band, and subsequently calculates smooth paths and velocities to estimate the expectations. Nevertheless, they assume that surrounding vehicles follow a constant speed in the sampled trajectories. This drawback is addressed by Huang, Wu and Lv (2021), which apply the Intelligent Driving Model (IDM) (TREIBER; HENNECKE; HELBING, 2000) to model vehicles' behavior by explicitly accounting for interaction between them. The trajectories are sampled via a short-term planner using polynomial curves. Although considering interaction is closer to real scenarios, polynomial curves are sometimes a simplistic assumption for imitating complex behaviors. Moreover, the authors assume full knowledge about vehicles' motion that are not overridden by IDM, which are considered to follow their original trajectories in the dataset. In our previous work (SILVA; GRASSI; WOLF, 2023), we tackle this problem by employing a Monte Carlo Tree Search (MCTS) to solve an interaction-aware MDP model, which can capture the interaction between surrounding vehicles and also incorporate noise in their future motion to account for uncertainty.

Nevertheless, in (SILVA; GRASSI; WOLF, 2023) we consider that surrounding vehicles behavior are completely determined by IDM, which cannot be assumed in a diverse and dynamic real traffic environment. For instance, human drivers might or might not give the right of way to other vehicles to perform a merging maneuver. Unfortunately, this behavior cannot be modeled

using only a following-car model such as IDM. This drawback can be overcome by considering that vehicles have its own particular behavior determined by their *internal intentions*, which cannot be directly observed.

6.2.2 IRL for POMDPs

The main consequence of such assumption is the need of employing methods that can handle partially observable environments, since other vehicles' internal intentions cannot be observed by the ego-vehicle in the absence of vehicle-to-vehicle communication (V2V). In this regard, Partially Observable Markov Decision Process (POMDP) becomes a natural choice to deal with such problem. Although POMDP models has been widely studied in autonomous driving field (LIU *et al.*, 2015; SEZER *et al.*, 2015; SUNBERG; HO; KOCHENDERFER, 2017; HUBMANN *et al.*, 2018a; HUBMANN *et al.*, 2018b), the ego-vehicle behavior is commonly determined by a reward function that combines features and hand-craft tuned weights. To the best of our knowledge, few works address the problem of IRL for POMDPs, which are discussed in the remaining of this section.

POMDPs are computationally hard to solve, making it difficult to be directly applied in IRL algorithms. Choi and Kim (2011) were the first authors to deal with this problem. They approximate the expert belief trajectories by using a Finite State Control (FSC). After, they use a point-based policy iteration (JI *et al.*, 2007) as POMDP solver to find an approximate optimal FSC policy on the reachable beliefs. Then, they extend maximum-margin-based approaches for MDPs (NG; RUSSELL *et al.*, 2000) to POMDP domains. Similarly, Chinaei and Chaib-draa (2014) propose methods to approximate the transition function of the expert belief trajectories as well as the use of a point-based value iteration algorithm (SPAAN; VLASSIS, 2005) to alleviate the computation burden of solving intermediate policies. However, both approaches require solving a POMDP in the forward problem at each update step of the reward weights. In this sense, Hussein, Begum and Petrik (2019) approximates the POMDP problem to an MDP either by directly mapping a set of observations to a specific state or by the discretization of the POMDP belief to a finite number of segments. Nevertheless, by approximating the POMDP model by an MDP model, the gain knowledge after making subsequent observations might be lost during the process. Moreover, the aforementioned works uses maximum-margin-based methods to learn the reward function, which cannot handle the ambiguity caused by sub-optimal demonstrations in contrast to maximum entropy-based approaches. By addressing those problems, Djeumou *et al.* (2022) propose an Sequential Convex Programming formulation to solve the forward POMDP problem in large state and observation spaces. They also apply a maximum causal-entropy approach to deal with sub-optimal demonstrations. However, they present results in classical benchmarks, which only consider discrete state spaces as well as synthetic data. In our previous work (SILVA; GRASSI; WOLF, 2019), we apply IRL directly on continuous-state POMDP for autonomous driving domain, but we also only consider synthetic data coming from an autopilot

simulator in a simplistic scenario.

In contrast to the aforementioned works, in this chapter we apply maximum entropy IRL to a continuous-state POMDP model in a highly interactive driving scenario using real data. By using an online POMDP solver, we estimate the feature expectation by sampling trajectories in regions with higher rewards, as an extension of our previous work (SILVA; GRASSI; WOLF, 2023). The proposed POMDP model can estimate surrounding vehicles intentions, which improves the computed policies according to the current rewards. This leads to a better learning performance when compared to MDP approaches, since the feature expectation estimation is directly affected by the chosen sampler.

6.3 Methodology

The approach employed to learn expert behavior in partially observable environments proposed in this chapter is similar to the one described in Section 5.3. The IRL strategy applied to continuous state spaces is the same as presented in Section 5.3.1, therefore it is omitted here for brevity.

6.3.1 POMDP Sampler

In the remaining of this section, we describe the key components of the POMDP model proposed in this chapter.

6.3.1.1 State Space

The state space $\mathcal{S}$ includes the states of all vehicles considered in the scene,

$$\mathcal{S} = [s_0 \quad s_1 \quad s_2 \quad \cdots \quad s_K]^{\mathrm{T}}, \tag{6.1}$$

where s_0 represents the state of the ego-vehicle and $s_k \in \{1, \cdots, K\}$ are the states of surrounding vehicles. For simplification, only four vehicles are considered in the model, according to Fig. 37: the back-vehicle, which is following the ego-vehicle on the same lane; the host-vehicle is the one immediately behind the ego-vehicle on the target lane; and the vehicles in front, which are being followed by the ego-vehicle on both current and target lanes.

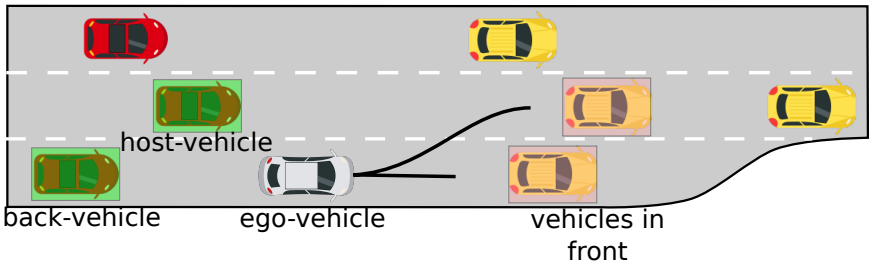
The state of all vehicles, with the exception of the host-vehicle state, is defined as

$$s_k = [d_k \quad u_k \quad v_{d\_k}]^{\mathrm{T}}, \quad s_k \neq s_{host}, \tag{6.2}$$

where d_k is the longitudinal distance from the start of the road to the vehicle center; u_k is the lateral distance from the center of the right-most lane to the vehicle center; and v_{d_k} is the longitudinal speed. Similarly, the state of the host-vehicle is defined as

$$s_{host} = [d_{host} \quad u_{host} \quad v_{d\_host} \quad \beta]^{\mathrm{T}}. \tag{6.3}$$

Figure 37 – Description of the vehicles in the state space: only the ego-vehicle, host-vehicle, back-vehicle and vehicles in front are considered in the planning.



Source: Elaborated by the author.

The state variable $\beta \in \{0, 1\}$ is introduced to model whether the host-vehicle is acting cooperatively ($\beta = 1$) or not ($\beta = 0$). In this work, to be *cooperative* means that the intention of the host-vehicle is to give the right of way to the ego-vehicle to perform the lane change. Therefore, β is an internal state that cannot be directly observed by the ego-vehicle.

6.3.1.2 Action Space and Transition Model

The transition on the state variables related to the vehicles' motion is constrained to the following motion model:

$$\begin{bmatrix} d'_k \\ v'_{d\_k} \\ u'_k \end{bmatrix} = \begin{bmatrix} d_k \\ v_{d\_k} \\ u_k \end{bmatrix} + \begin{bmatrix} v\_d_k \\ a_{d\_k} \\ v_{u\_k} \end{bmatrix} \Delta t + \frac{1}{2} \begin{bmatrix} a_{d\_k} \\ 0 \\ 0 \end{bmatrix} \Delta t^2, \tag{6.4}$$

in which a_{d_k} and v_{u_k} is the speed rate change and lateral speed of vehicle k, respectively, and Δt is the time step. The ego-vehicle's longitudinal acceleration a_{d_0} is chosen based on UCT criterion, where a_{d_0} can be selected from $\mathcal{A}_{a_d_0} = \{-0.7,\ 0,\ 0.7\}$m/s. The back-vehicle is assumed to plan its speed according to the IDM model (TREIBER; HENNECKE; HELBING, 2000):

$$a_{IDM} = a_{max} \left(1 - \left(\frac{v_k}{v_{desired}} \right)^{\delta} - \left(\frac{g^*(v,\ \Delta v)}{g} \right)^2 \right) \tag{6.5}$$

$$g^*(v,\ \Delta v) = g_{desired} + t_{tc} v + \frac{v \Delta v}{2\sqrt{a_{max} b_{comfort}}}, \tag{6.6}$$

where the rear-to-front distance g and the speed difference Δv are computed with respect to either the vehicle in front or end of its current lane. Also, δ is the acceleration exponent, a_{max} is the desired maximum acceleration, $b_{comfort}$ is the comfort deceleration, $v_{desired}$ is the desired speed and t_{tc} is the desired time gap.

The behavior of the host-vehicle depends on its internal state β, which is considered constant during the planning ($\beta' = \beta$). When $\beta = 0$, the host-vehicle is assumed to act egoistically and does not give the right of way to the ego-vehicle by behaving according to IDM, where

$\{a_{d\_host} = a_{IDM} \,|\beta = 0\}$. However, it may cooperate with the ego-vehicle by decelerating and giving room for a lane change, in which we consider $\{a_{d\_host} = -1\text{m/s}^2 \,|\beta = 1\}$. It is important to note that, after completing the lane change, the host-vehicle becomes the new back-vehicle, since it and the ego-vehicle are on the same lane. Additionally, the vehicles in front are assumed to follow a constant speed by making $a_{d\_front} = 0$. In order to account for model uncertainties and physical constraints,

$$a_{d\_k} = \max\Big(a_{pred} + \mathcal{N}(0, \sigma^2_{acc}),\ -b_{safe}\Big),\ \forall k \neq 0 \tag{6.7}$$

where $\mathcal{N}(0, \sigma^2_{acc})$ is a zero mean Gaussian noise acceleration, b_{safe} is the maximum safe deceleration, and a_{pred} is the predicted acceleration according to the considered vehicle.

The lateral position of surrounding vehicles is considering constant by making $\{v_{u\_k} = 0 \mid \forall k \neq 0\}$. As in Hubmann *et al.* (2018b), we consider that the ego-vehicle can instantaneously change its lateral speed v_{u_0} according to $\mathcal{A}_{0\_lat} = \{LC,\ SCL\}$, where:

$$\begin{cases} v_{u\_0} = & \min(0.17 v_{d\_k},\ 0.6), & \text{if } a_{0\_lat} \text{ is LC} \\ v_{u\_0} = & 0, & \text{otherwise} \end{cases}, \tag{6.8}$$

in which LC and SCL represent a change to the target lane, and staying in the current lane, respectively.

6.3.1.3 Observation Space and Particle Filter

The ego-vehicle's perception system gives the position and longitudinal speed of all vehicles considered in the scene. However, it cannot determine whether the host-vehicle is cooperating with the ego-vehicle or not. Therefore, the ego-vehicle must infer it by gathered observations.

The observation space $\mathcal{O}$ is defined as

$$\mathcal{O} = [o_0 \quad o_1 \quad o_2 \quad \dots \quad o_K]^{\mathrm{T}}, \tag{6.9}$$

where o_0 represents the observation of the ego-vehicle state and $o_k \in \{1, \cdots, K\}$ are the observation of surrounding vehicles states:

$$o_k = [d_{\text{obs}_k} \quad u_{\text{obs}_k} \quad v_{\text{obs}_{kd\_k}}]^{\mathrm{T}}. \tag{6.10}$$

POMCP samples discrete observations according to a predefined observation model. Hence, the continuous observation o_k must be converted to a discrete one in order to expand the belief tree. To accomplish that, we model the observation as a noisy measurement of β given by a weighted particle filter. Similarly to Sunberg, Ho and Kochenderfer (2017), the particles are weighted by considering that the observed speed of the host-vehicle $v_{\text{obs\_}host}$ is sampled from a Gaussian distribution:

$$P(o_{host}|s'_{host}) \propto \exp\Big(-\frac{(v_{\text{obs\_}host} - v_{pred})^2}{2\sigma^2_{vel}}\Big)$$

where σ_{vel}^2 is the standard deviation of the distribution, and v_{pred} is the predicted host-vehicle speed according to the transition model and β. This simplification reduces the observation space to $o_{host} \in \{0,\ 1\}$, thus avoiding the need of using continuous observation POMDP solvers, such as POMCPOW (SUNBERG; KOCHENDERFER, 2018).

6.3.1.4 Reward Model

As discussed in Section 5.3.1, the reward model $R_\theta(s,\ a)$ is a linear combination of features $\mathbf{f}_s$ and a learned set of weights θ. The features must encode important aspects of human driver behaviors so that the reward model be capable of imitating their trajectories. The set of chosen features are the same as presented in Section 5.4.3.

6.3.1.5 Deterministic Constant Action Heuristic

In order to speed up the optimal policy convergence, a deterministic heuristic is applied to POMCP at each time an episode h in $\mathcal{T}$ is concluded. An episode is composed by a sequence of states, actions, observations and rewards sampled from the generative model, and is finished once a new node is reached. The estimated value at the leaf node can be considered as the total sum of discount rewards computed from the reached state to the maximum length of the belief tree $H_\mathcal{T}$, and by assuming that the problem is fully observable. In an intuitive way, the estimated value gives the potential of exploration of each node, speeding up the policy convergence.

Instead of using a rollout policy based on uniform random action selection as heuristic (SILVER; VENESS, 2010), we consider that the ego-vehicle executes the last sampled action in the subsequent time steps until $H_\mathcal{T}$ is reached. This assumption is justified by the fact that human drivers avoid changing acceleration and lateral speed to improve comfort. Therefore, the same behavior must be observed during the learning of $R_\theta(s,\ a)$. It should be noted that this simplification on the heuristic is made in order to diminish the computational complexity, since it must be solved at each new node of $\mathcal{T}$. The more complex the heuristic is, the more time is required to simulate a complete rollout, which can directly affect the quality of the solution.

6.3.2 Algorithm Summary

The summary of the proposed Maximum Entropy IRL via POMDP trajectory sampling algorithm (MEIRL-POMDP) can be seen in Algorithm 2.

6.4 Experiment Setup

6.4.1 Dataset

The training data are selected from the subset DR_DEU_Merging_MT belonging to the INTERACTION dataset (ZHAN *et al.*, 2019), as shown in Fig. 38. The dataset provides

Algorithm 2: Maximum Entropy Inverse Reinforcement Learning via POMDP trajectory sampling

Input : Human drivers demonstrations $\mathcal{D}$, environment model $\mathcal{C}$, interaction-aware POMDP model, POMCP online solver, learning rate α, regularization parameter λ, number of iterations I and number of epochs E.

Result: optimized reward weights θ^*

Initialize $\theta \leftarrow random([-1, 1])$;
Compute feature expectation for expert demonstrations $\bar{\mathbf{f}}(\mathcal{D}) = \frac{1}{M}\sum_{\zeta\in\mathcal{D}} \mathbf{f}_\zeta$;
Initialize the samples dataset $\tilde{\mathcal{D}}_{i=0:M} \leftarrow [\,]$;
Initialize the reward weights buffer $\Theta \leftarrow [\,]$;
for $iteration \leftarrow 1$ *to* I **do**
  foreach $\zeta_i \in \mathcal{D}$ **do**
    Generate a trajectory $\tilde{\zeta}_i$ with the same initial conditions as ζ_i using the POMDP model, the POMCP online solver and environment model $\mathcal{C}$;
    Add $\tilde{\zeta}_i \rightarrow \tilde{\mathcal{D}}_i$;
  end
  Add $\theta \rightarrow \Theta$;
  For each initial condition, calculate $q_j(\tilde{\zeta}_i) = \frac{\exp(R_{\theta_j}(\tilde{\zeta}_i))}{\sum_{i=0}^{|\Theta|}\exp(R_{\theta_j}(\tilde{\zeta}_i))}$ for each $\theta_j \in \Theta$;
  Compute the importance weights $z(\tilde{\zeta}_i) = \frac{1}{|\Theta|}\sum_{j=0}^{|\Theta|}\frac{1}{q_j(\tilde{\zeta}_i)}$;
  for $epoch \leftarrow 1$ *to* E **do**
    Compute the features expectation with the sampled trajectories for each initial condition: $\tilde{\mathbf{f}} = \frac{1}{Z}\sum_{\tilde{\zeta}_{i=0}}^{|\tilde{\mathcal{D}}_i|} w_{\tilde{\zeta}_i}\mathbf{f}_{\tilde{\zeta}_i}$;
    Calculate the gradient $\nabla_\theta\mathcal{L}(\theta) = \frac{1}{M}\sum_{i=0}^{M}\left[\bar{\mathbf{f}}_i(\mathcal{D}) - \tilde{\mathbf{f}}_i(\tilde{\mathcal{D}})\right] - 2\lambda\theta$;
    Update the reward weights $\theta \leftarrow \theta + \alpha\nabla_\theta\mathcal{L}(\theta)$;
  end
end

the position in global coordinates, the heading and the speed of the vehicles in the scene. This information is used to reconstruct the road structure throughout a whole trajectory. The duration of each trajectory is 4s duration and the sampling time $\Delta t_{sim} = 0.1$s. The chosen merging scenario is composed of two lanes: the length of the right lane and the left lane are 55m and 110m, respectively, and the width of both lanes is 3.0m, approximately. This means that the vehicles on the right must change lane to continue its travel. Finally, 100 trajectories are selected as training data and 50 trajectories as test data. The trajectories are selected from different locations in the scene to provide a diverse training and test data.

6.4.2 Implementation Details

POMCP and the POMDP model are implemented using *POMDP.jl* (EGOROV *et al.*, 2017), a framework for sequential decision making under uncertainty. *POMDP.jl* is implemented in *Julia*, a high-level, dynamic and high-performance language. On the other hand, the environ-

ment model required to simulate the sampled trajectories, and also the Maximum Entropy IRL algorithm are implemented using *Python*. During the simulation, vehicles on the same lane and behind the ego-vehicle are overridden by IDM to simulate a more realistic interaction behavior. The values of the IDM parameters are $\delta = 4$, $g_{desired} = 1.0$m, $t_{tc} = 1.0$s, $a_{max} = 1.0$m/s^2, $b_{comfort} = 2.0$m/s^2, $v_{desired} = 11.0$m/s and $b_{safe} = 4$m/s^2. The other vehicles follow their original trajectories in the dataset. The system containing the simulation environment and algorithms runs on a Intel Core i7-6500U CPU CPU with 2.50 Hz, a 7.7 GB memory RAM and a NVIDIA GeForce 930M. The code is available on https://bitbucket.org/juniorars/pomdp_irl/src/master/.

The main POMDP model and POMCP parameters are $c = 1.0$, $\gamma = 0.99$, $\Delta t = 0.8$s, $\sigma^2_{acc} = 0.05$m/s^2, $\sigma^2_{vel} = 0.3$m/s and $H_{\mathcal{T}} = 5$. The IDM parameters used in the POMDP model are the same as of the simulation. Additionally, we need to smooth the computed trajectory, since the simulation runs at a higher frequency than POMCP. Therefore, we employ a ramp-based trajectory planning for longitudinal acceleration and lateral speed according to the their current values as well as the target values computed by the solver. The algorithm runs for 10 iterations, where the reward weights are trained using 1e4 epochs with a L2 normalization parameter $\lambda = 0.001$ and learning rate $\alpha = 0.05$.

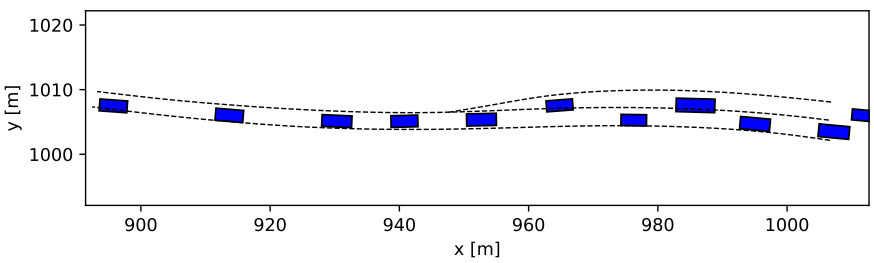
6.4.3 Baseline methods

To validate our method, we compare the POMDP approach with different models that do not consider partially observable states. Those models differ in the way they predict the host-vehicle behavior and are described below:

6.4.3.1 MDP-IDM

This model considers that the host-vehicle always behaves according to IDM. Therefore, it follows the vehicle in front and does not act cooperatively with the ego-vehicle.

Figure 38 – DR_DEU_Merging_MT in the INTERACTION dataset. The vehicles are moving from right to left.



Source: Elaborated by the author.

6.4.3.2 MDP Courteous

Unlike the previous model, the MDP Courteous assumes that the host-vehicle always act cooperatively and gives the right of way to the ego-vehicle until it reaches the target lane. After that, the host vehicle behaves according to IDM.

6.4.3.3 MDP Constant Speed

In this model, the host vehicle is considered to follow a constant speed until the ego-vehicle changes lanes. Subsequently, it starts to behave according to IDM.

6.4.3.4 MDP without interaction

It is also important to analyze the effects of the interaction between the ego and other vehicles. In this sense, we employ an MDP model in which all surrounding vehicles are assumed to follow a constant speed. As a result, this model lacks the ability of predicting surrounding vehicles' deceleration.

To make a fair comparison, the baselines are used to learn their own set of reward weights θ^* using Algorithm 2. Moreover, the set of learning and model parameters are the same as the ones employed in the POMDP approach as well as the constant action heuristic.

6.4.4 Evaluation Metrics

The main metrics used to compare the different approaches are feature deviation and mean Euclidean distance.

6.4.4.1 Feature deviation

The goal of IRL is to imitate expert demonstrations by computing a feature distribution similar of the one in the dataset. The feature deviation (FD) can be computed as

$$\mathcal{E}_{FD} = \frac{1}{M}\sum_{i=1}^{M}\frac{1}{N_i}|\mathbf{f}(\zeta_i^{gt}) - \mathbf{f}(\zeta_i^{plan})|, \tag{6.11}$$

where M is the number of expert demonstrations and N_i is the length of the i-th trajectory.

6.4.4.2 Mean Euclidean Distance (MED)

The Mean Euclidean Distance (MED) is computed by considering the final position of the ego-vehicle for a trajectory in the demonstration and the trajectory given by the sampler with the same initial conditions:

$$\mathcal{E}_{MED} = \frac{1}{M}\sum_{i=1}^{M}||\zeta_i(N_i)^{gt} - \zeta_i(N_i)^{plan}||_2. \tag{6.12}$$

6.4.4.3 Hard Deceleration

It is expected that the ego-vehicle navigates without causing discomfort for surrounding vehicles. This metric is calculated according to the number of time steps in which the ego-vehicle causes a hard deceleration $a_{hard} < -3.0\text{m/s}^2$ to other vehicles:

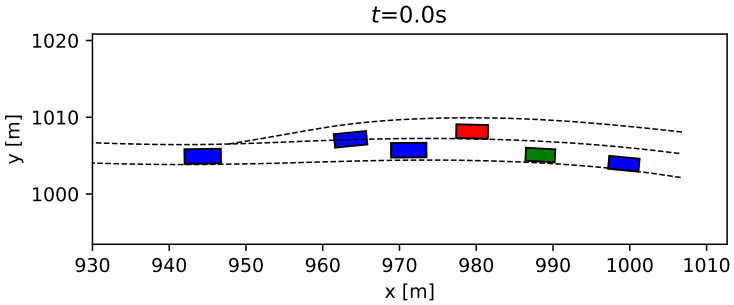
$$\mathcal{E}_{HD} = \sum_{i=1}^{M} \sum_{j=1}^{N} \left(a_{back}(N_j)^{plan} < a_{hard} \right). \tag{6.13}$$

6.5 Results

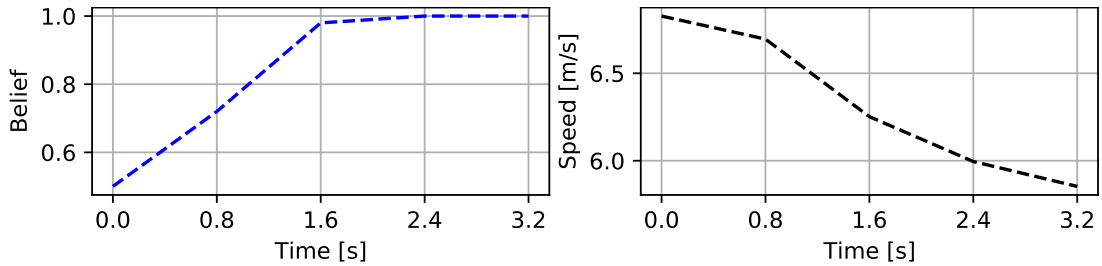
6.5.1 Host Vehicle Intention Estimation

The main reason of applying a POMDP to plan trajectories is its ability to estimate surrounding vehicles' internal states during the planning. In this section, we present three common scenarios encountered in merging maneuvers according to the host-vehicle behavior. It is important to highlight that the belief estimation in all scenarios detailed below is performed in real data according to the dataset.

Figure 39 – Example scenario in which the host-vehicle decelerates in order to cooperate with the ego-vehicle.



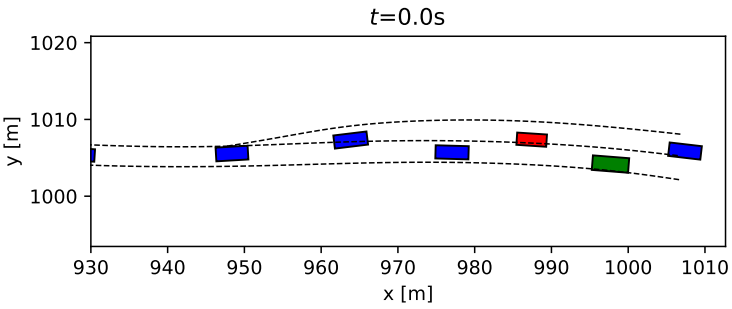
(a) Trajectory positions at $t = 0$s. The ego-vehicle is colored in red and the host-vehicle is in green. The remaining vehicles are in blue.



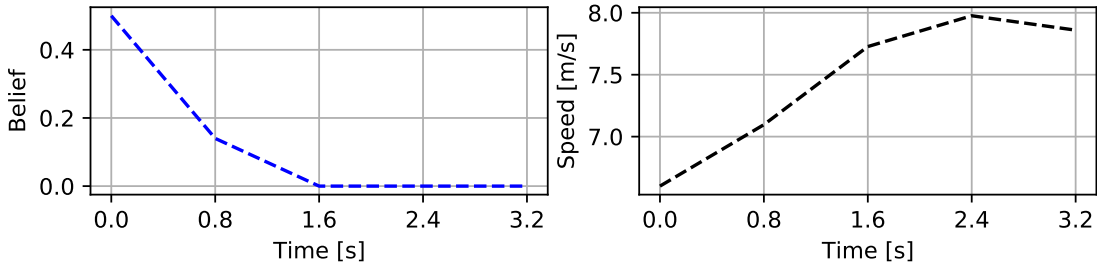
(b) Estimation of belief about β (left) according to the host-vehicle speed (right).

Source: Elaborated by the author.

Figure 40 – Example scenario in which the host-vehicle does not give the right of way to the ego-vehicle.



(a) Trajectory positions at $t = 0$s. The ego-vehicle is colored in red and the host-vehicle is in green. The remaining vehicles are in blue.



(b) Estimation of belief about β (left) according to the host-vehicle speed (right).

Source: Elaborated by the author.

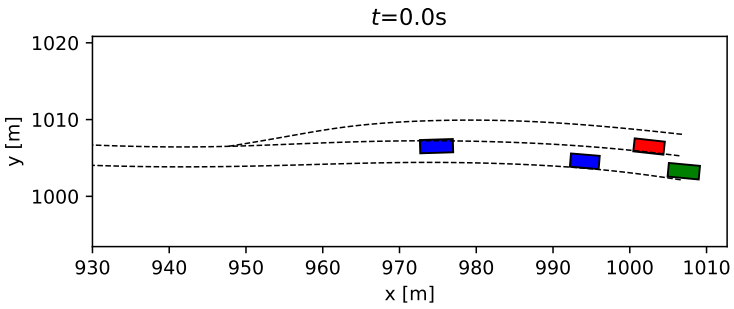
6.5.1.1 Giving the right of way

Fig. 39 exemplify a situation where the ego-vehicle is approaching the end of its current lane (Fig. 39a). As we can seen in Fig. 39b, the host-vehicle starts decreasing its speed with respect to time, which increases the current belief about the host-vehicle cooperativeness. At $t = 1.6$s, after two observations, the belief almost converge to $\beta = 1$. Estimating that the host-vehicle is giving the right of way is a valuable information used by the ego-vehicle to plan a lane change maneuver without forcing the host-vehicle to perform a hard deceleration.

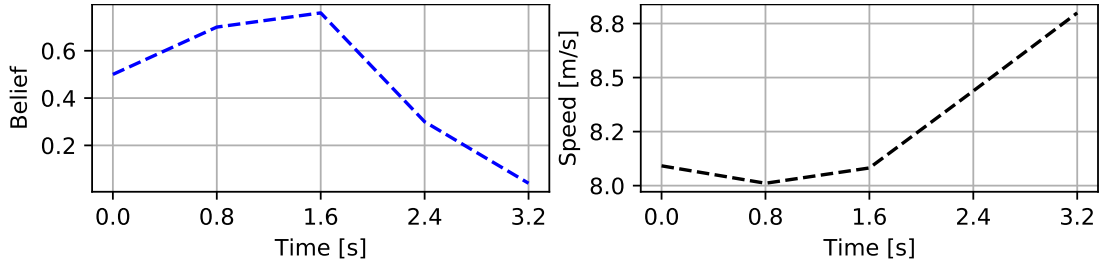
6.5.1.2 Non-cooperative behavior

Sometimes, depending on the position of the ego-vehicle along the road and the gap on the target lane, the host-vehicle might decide not to giving the right of way. One example of this situation is depicted in Fig. 40. At the beginning, the host vehicle is traveling with a speed near to 6.5 m/s. After making the first observation and perceiving that the host-vehicle speed is increasing, the belief about the $\beta = 1$ decreases below to 0.2. At the second observation, the belief converges to $\beta = 0$, meaning that the host-vehicle does not intend to give the right of way to the ego-vehicle. With this information, the ego-vehicle must decide whether to stay on its current lane or to increase its speed to avoid forcing the host-vehicle to decelerate abruptly.

Figure 41 – Example scenario in which the belief about the host-vehicle intention changing after subsequent observations.



(a) Trajectory positions at $t = 0$s. The ego-vehicle is colored in red and the host-vehicle is in green. The remaining vehicles are in blue.



(b) Estimation of belief about β (left) according to the host-vehicle speed (right).

Source: Elaborated by the author.

6.5.1.3 Changing behavior estimation

The ego-vehicle must also be aware about changes on the host-vehicle behavior, as shown in Fig. 41. At first, the host-vehicle seems to decrease its speed in order to act cooperatively with the ego-vehicle. As a consequence, the belief about $\beta = 1$ increases. However, at $t = 1.6$s the host vehicle starts accelerating, which causes a change in the belief. One possible reason for the change in the belief is that the host vehicle initially decelerates in order to keep a safe distance to the vehicle in front. As a result, this action is misinterpreted as an attempt of cooperation. Nevertheless, this example shows the effectiveness and robustness of the belief estimation according to the observation uncertainty introduced by σ_{vel}, which prevents β to converge at the beginning of the filtering process.

6.5.2 Performance on the Test Set

This section presents the results on the test set of our proposed method as well as of the baselines described in Section 6.4.3. Since POMCP is an any-time, sample-based solver, we run the experiments for three times and then the mean of the metrics are calculated, as shown in Table 12. We found that three runs of experiments are enough as the solver produces very similar

Table 12 – Results on the test set.

	f_v	f_m	f_i	f_g	f_j	f_l	f_c	MED (m)	Hard Deceleration
POMDP	0.025± 0.034	0.042± 0.048	0.047± 0.26	0.036± 0.066	0.072± 0.11	0.15± 0.27	0.000± 0.000	1.37± 0.94	7.67
MDP-IDM	0.026± 0.036	0.043± 0.049	0.047± 0.27	0.037± 0.065	0.073± 0.10	0.16± 0.24	0.000± 0.000	1.48± 1.01	8.33
MDP Courteous	0.028± 0.039	0.043± 0.048	0.050± 0.28	0.037± 0.064	0.074± 0.11	0.16± 0.24	0.000± 0.000	1.54± 1.05	9.67
MDP Const. Speed	0.028± 0.038	0.044± 0.049	0.051± 0.30	0.036± 0.062	0.075± 0.11	0.15± 0.23	0.000± 0.000	1.59± 1.11	10.67
MDP w/o inter.	0.028± 0.038	0.044± 0.049	0.049± 0.27	0.037± 0.064	0.075± 0.12	0.16± 0.24	0.000± 0.000	1.58± 1.16	9.00

behaviors at each run.

6.5.2.1 Feature Deviation

As we can see in Table 12, our proposed method presents the best results when compared to the baselines, except for f_l. The probable reason for this result is that the POMDP sampler can plan better trajectories according to the current rewards, because of its ability to estimate the host-vehicle intention. The main consequence is the achievement of learned weights that better describe human drivers behavior, even on the test set.

6.5.2.2 Mean Euclidean Distance (MED)

Our method also present the best result when the MED is considered. This is in accordance with the feature deviation metric, showing that the chosen features can capture the human driving styles present in the dataset.

6.5.2.3 Hard Deceleration

This result shows that the POMDP planner is the best method to avoid causing back vehicles hard deceleration when compared to the baselines. This result is important since one of the main goal of autonomous vehicles is to join another vehicles in a smooth way, without causing discomfort to them.

It important to note that, at first glance, the results seem to be very similar among all methods. However, as stated in Section 6.4.1, all trajectories in the dataset have a duration of 4s. As a consequence, the planner can compute only four actions according to $\Delta t = 0.8$. The results difference would probably be more significant if a dataset with longer trajectories were considered.

6.5.3 The Effect of Heuristic

In this section, we analyze the effect of the constant action heuristic on the ego-vehicle learned behavior when compared to a random rollout policy heuristic. The two models with better results in the previous section are chosen in this analysis, which can be seen in Table 13.

Table 13 – Results of the POMDP and MDP-IDM models with and without constant action heuristic.

	f_v	f_m	f_i	f_g	f_j	f_l	f_c	MED (m)	Hard Deceleration
POMDP w/ heur.	0.025± 0.034	0.042± 0.048	0.047± 0.26	0.036± 0.066	0.072± 0.113	0.15± 0.27	0.000± 0.000	1.37± 0.94	7.67
POMDP w/o heur.	0.028± 0.036	0.043± 0.048	0.058± 0.31	0.038± 0.065	0.076± 0.154	0.15± 0.24	0.000± 0.000	1.66± 1.18	12.00
MDP-IDM w/ heur.	0.026± 0.036	0.043± 0.049	0.047± 0.27	0.037± 0.065	0.073± 0.105	0.16± 0.24	0.000± 0.000	1.48± 1.02	8.33
MDP-IDM w/o heur.	0.027± 0.039	0.043± 0.048	0.051± 0.28	0.037± 0.063	0.066± 0.100	0.16± 0.24	0.000± 0.000	1.52± 1.04	10.00

The results show that the performance of both methods drops performance without our proposed heuristic. However, the POMDP approach is more affected than the MDP-IDM when the heuristic is not applied, presenting the worse results even when compared to other methods in Table 12. One possible reason is that POMDPs presents high computation complexity, since the belief tree are expanded according to chosen actions and also sampled observations. As a consequence, this complexity prevents the POMDP solver to converge to good policies when a proper heuristic is not employed. Therefore, this result shows the effectiveness of our proposed heuristic to estimate the value at leaf nodes of the belief tree.

6.6 Final Considerations

This chapter presents a variation of the Maximum Entropy IRL algorithm that uses a POMDP online solver to sample trajectories in order to estimate the gradients of reward weights. By modeling the IRL problem as a POMDP, we achieves a better learning performance compared to baselines that consider a fully observable state space. This result is highly impacted by the POMDP model capability of estimating host-vehicles intention during merging maneuvers. Moreover, the presented constant action heuristic improves the POMDP performance by alleviating the computation complexity. As a result, our MEIRL-POMDP algorithm outperformed IRL approaches that are modeled as an MDP problem.

CHAPTER

7

CONCLUSION

In this thesis, we have proposed a diversity of models capable of dealing with the uncertainty inherent to decision-making for autonomous vehicles. The proposed models are based on specific urban scenarios encountered in the autonomous driving field. Moreover, we have presented algorithms capable of automatically learning human driving behaviors from expert human trajectories. In this chapter, we analyze the contributions of each method proposed throughout this thesis and propose future work to improve each approach as well.

Chapter 3 presents a POMDP model to compute the acceleration of autonomous vehicle at signalized intersections, in which the information about the traffic light color is assumed as partially observable. The model is able to produce satisfactorily results in different initial conditions and proved to perform better than approaches which do not consider the erroneous measurements coming from sensors. For future work, the authors intend to evaluate the duration of each phase (color) using data from real traffic light systems during different periods of the day as well as diverse traffic conditions in order to improve the robustness of the proposed model.

Chapter 4 details a POMDP model to calculate longitudinal and lateral actions on multi-lane roads in the presence of surrounding vehicles. The POMDP model uses sensor information to estimate ongoing lane changes performed by other vehicles, which aided the planning of safe trajectories. Moreover, a motion model is proposed in order to predict surrounding vehicles speed during the interaction with the ego and other vehicles. For future work, a model considering different behavior during overtaking, such as velocity increasing, will be pursued. The model might assume a target speed during overtaking or even a difference between the ego-vehicle speed and the speed of the vehicle being overtaken.

Chapter 5 presents a variation of the Maximum Entropy IRL to deal with continuous state spaces as well as interaction between vehicles in a merge scenario. A trajectory sampling approach is proposed in order to adapt the sampling strategy to regions of greater rewards, which guides the reward function to the optimality according to the expert demonstrations. The

results show that our method outperformed models such as IDM and MOBIL as well as typical polynomial curve-based sampling for IRL.

Chapter 6 shows an extension of the sampling strategy proposed in Chapter 5, in which more complex behaviors of surrounding vehicles during merging is assumed. In this regard, a POMDP model is presented, which considers the unobserved intention of surrounding vehicles of giving (or not giving) the right of way to the ego-vehicle. This assumption proved to improve the performance of the learning algorithm when compared to deterministic approaches that consider that the behavior of surrounding vehicles are fully observable. Currently, our method assumes a simple ramp-based trajectory planner to smooth the actions planned by the POMDP solver. This might sometimes lead to unnecessary control efforts and poor smooth trajectories, since the resulting longitudinal acceleration and lateral speed curves does not have higher derivatives. Therefore, the employment of more refined trajectory planners will be seek in future work. Additionally, the model assumes perfect knowledge of IDM parameters of the vehicles overridden by the simulator. In future work, we intend to analyze how the robustness of the model is affected by the chosen parameters. Moreover, the chosen features have great impact on the learned behavior, thus it is difficult to know whether those features are the best choice. In this regard, methods using deep learning approaches in which the features can be automatically learned by a neural network will be pursued.

Although the combination of IRL and POMDP (MDP) methods has presented satisfactorily results in the proposed scenarios, a point of attention is that those methods are very computational complex. Moreover, each model is dedicated to one specific scenario, which makes it difficult to extrapolate the models to unseen environments. Therefore, more studies might be carried on to know whether the proposed methods are applicable in real-world situations. For future work, we intend to explore models that can be generalized easier to other domains. One potential improvement might be to learn a generic dynamic model from data using deep learning and then learning the underlying reward function through IRL. Moreover, we intend to implement the solutions in the autonomous vehicle CaRINA II to verify their effectiveness in real domains. This vehicle is a Fiat Palio Adventure Dualogic, adapted with sensors and actuators that enable autonomous operation. CaRINA II's software architecture is implemented using Robot Operating System (ROS), a collection of software frameworks for robotic systems development. CARINA II has a perception module used to detect static and moving obstacles along with path planning and control algorithms, which will be used to generate low level signals (steering angle and acceleration) from the decision making module. The tests are intended to be made in a controlled environment in order to avoid dealing with dangerous situations.

BIBLIOGRAPHY

ASADI, B.; VAHIDI, A. Predictive cruise control: Utilizing upcoming traffic signal information for improving fuel economy and reducing trip time. **IEEE transactions on control systems technology**, IEEE, v. 19, n. 3, p. 707–714, 2010. Citation on page 44.

BANDYOPADHYAY, T.; WON, K. S.; FRAZZOLI, E.; HSU, D.; LEE, W. S.; RUS, D. Intention-aware motion planning. In: **Algorithmic foundations of robotics X**. [S.l.]: Springer, 2013. p. 475–491. Citation on page 58.

BEHRENDT, K.; NOVAK, L.; BOTROS, R. A deep learning approach to traffic lights: Detection, tracking, and classification. In: IEEE. **2017 IEEE International Conference on Robotics and Automation (ICRA)**. [S.l.], 2017. p. 1370–1377. Citation on page 44.

BELLMAN, R. A markovian decision process. **Journal of Mathematics and Mechanics**, JSTOR, p. 679–684, 1957. Citation on page 34.

BELLMAN, R. E. **Dynamic programming**. [S.l.]: Princeton university press, 2010. Citation on page 45.

BEVLY, D.; CAO, X.; GORDON, M.; OZBILGIN, G.; KARI, D.; NELSON, B.; WOODRUFF, J.; BARTH, M.; MURRAY, C.; KURT, A.; REDMILL, K.; OZGUNER, U. Lane change and merge maneuvers for connected and automated vehicles: A survey. **IEEE Transactions on Intelligent Vehicles**, IEEE, v. 1, n. 1, p. 105–120, 2016. Citation on page 56.

BOJARSKI, M.; TESTA, D. D.; DWORAKOWSKI, D.; FIRNER, B.; FLEPP, B.; GOYAL, P.; JACKEL, L. D.; MONFORT, M.; MULLER, U.; ZHANG, J. *et al.* End to end learning for self-driving cars. **arXiv preprint arXiv:1604.07316**, 2016. Citation on page 29.

BOULARIAS, A.; KOBER, J.; PETERS, J. Relative entropy inverse reinforcement learning. In: JMLR WORKSHOP AND CONFERENCE PROCEEDINGS. **Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics**. [S.l.], 2011. p. 182–189. Citations on pages 76 and 78.

BRECHTEL, S.; GINDELE, T.; DILLMANN, R. Probabilistic mdp-behavior planning for cars. In: IEEE. **2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)**. [S.l.], 2011. p. 1537–1542. Citations on pages 33 and 79.

____. Probabilistic decision-making under uncertainty for autonomous driving using continuous pomdps. In: IEEE. **17th International IEEE Conference on Intelligent Transportation Systems (ITSC)**. [S.l.], 2014. p. 392–399. Citation on page 58.

BUEHLER, M.; IAGNEMMA, K.; SINGH, S. **The 2005 DARPA grand challenge: the great robot race**. [S.l.]: Springer, 2007. Citation on page 28.

CAMACHO, E. F.; ALBA, C. B. **Model predictive control**. [S.l.]: Springer science & business media, 2013. Citation on page 44.

CHEN, L.; ENGLUND, C. Cooperative intersection management: A survey. **IEEE Transactions on Intelligent Transportation Systems**, IEEE, v. 17, n. 2, p. 570–586, 2015. Citation on page 56.

CHINAEI, H.; CHAIB-DRAA, B. Dialogue pomdp components (part ii): learning the reward function. **International Journal of Speech Technology**, Springer, v. 17, p. 325–340, 2014. Citation on page 96.

CHOI, J.; KIM, K.-E. Inverse reinforcement learning in partially observable environments. **Journal of Machine Learning Research**, MICROTOME PUBL, v. 12, p. 691–730, 2011. Citation on page 96.

DJEUMOU, F.; CUBUKTEPE, M.; LENNON, C.; TOPCU, U. Task-guided inverse reinforcement learning under partial information. In: **Proceedings of the International Conference on Automated Planning and Scheduling**. [S.l.: s.n.], 2022. v. 32, p. 53–61. Citation on page 96.

DOSOVITSKIY, A.; ROS, G.; CODEVILLA, F.; LOPEZ, A.; KOLTUN, V. CARLA: An open urban driving simulator. In: **Proceedings of the 1st Annual Conference on Robot Learning**. [S.l.: s.n.], 2017. p. 1–16. Citations on pages 48 and 64.

EGOROV, M.; SUNBERG, Z. N.; BALABAN, E.; WHEELER, T. A.; GUPTA, J. K.; KOCHENDERFER, M. J. Pomdps. jl: A framework for sequential decision making under uncertainty. **The Journal of Machine Learning Research**, JMLR. org, v. 18, n. 1, p. 831–835, 2017. Citations on pages 82 and 101.

FERGUSON, D.; HOWARD, T. M.; LIKHACHEV, M. Motion planning in urban environments. **Journal of Field Robotics**, Wiley Online Library, v. 25, n. 11-12, p. 939–960, 2008. Citation on page 84.

FINN, C.; LEVINE, S.; ABBEEL, P. Guided cost learning: Deep inverse optimal control via policy optimization. In: PMLR. **International conference on machine learning**. [S.l.], 2016. p. 49–58. Citations on pages 77 and 78.

GINDELE, T.; BRECHTEL, S.; DILLMANN, R. Learning driver behavior models from traffic observations for decision making and planning. **IEEE Intelligent Transportation Systems Magazine**, IEEE, v. 7, n. 1, p. 69–79, 2015. Citations on pages 29 and 58.

GONZÁLEZ, D.; PÉREZ, J.; MILANÉS, V.; NASHASHIBI, F. A review of motion planning techniques for automated vehicles. **IEEE Transactions on intelligent transportation systems**, IEEE, v. 17, n. 4, p. 1135–1145, 2015. Citation on page 84.

GONZÁLEZ, D. S.; ERKENT, O.; ROMERO-CANO, V.; DIBANGOYE, J.; LAUGIER, C. Modeling driver behavior from demonstrations in dynamic environments using spatiotemporal lattices. In: IEEE. **2018 IEEE International Conference on Robotics and Automation (ICRA)**. [S.l.], 2018. p. 3384–3390. Citation on page 76.

HORST, R. V. D. *et al.* Driver decision making at traffic signals. **Transportation Research Record**, v. 1172, p. 93–97, 1988. Citation on page 43.

HUANG, X.; PENG, H. Speed trajectory planning at signalized intersections using sequential convex optimization. In: IEEE. **2017 American Control Conference (ACC)**. [S.l.], 2017. p. 2992–2997. Citation on page 44.

HUANG, Z.; WU, J.; LV, C. Driving behavior modeling using naturalistic human driving data with inverse reinforcement learning. **IEEE Transactions on Intelligent Transportation Systems**, IEEE, 2021. Citations on pages 76, 78, 83, 84, 85, and 95.

HUBMANN, C.; SCHULZ, J.; BECKER, M.; ALTHOFF, D.; STILLER, C. Automated Driving in Uncertain Environments: Planning With Interaction and Uncertain Maneuver Prediction. **IEEE Transactions on Intelligent Vehicles**, IEEE, v. 3, n. 1, p. 5–17, 2018. ISSN 2379-8904. Available: <http://ieeexplore.ieee.org/document/8248668/>. Citations on pages 57, 58, and 96.

HUBMANN, C.; SCHULZ, J.; XU, G.; ALTHOFF, D.; STILLER, C. A belief state planner for interactive merge maneuvers in congested traffic. In: IEEE. **2018 21st International Conference on Intelligent Transportation Systems (ITSC)**. [S.l.], 2018. p. 1617–1624. Citations on pages 58, 80, 96, and 99.

HUSSEIN, A.; GABER, M. M.; ELYAN, E.; JAYNE, C. Imitation learning: A survey of learning methods. **ACM Computing Surveys (CSUR)**, ACM New York, NY, USA, v. 50, n. 2, p. 1–35, 2017. Citation on page 74.

HUSSEIN, M.; BEGUM, M.; PETRIK, M. Inverse reinforcement learning of interaction dynamics from demonstrations. In: IEEE. **2019 International Conference on Robotics and Automation (ICRA)**. [S.l.], 2019. p. 2267–2274. Citation on page 96.

JENSEN, M. B.; PHILIPSEN, M. P.; MØGELMOSE, A.; MOESLUND, T. B.; TRIVEDI, M. M. Vision for looking at traffic lights: Issues, survey, and perspectives. **IEEE Transactions on Intelligent Transportation Systems**, IEEE, v. 17, n. 7, p. 1800–1815, 2016. Citation on page 44.

JI, S.; PARR, R.; LI, H.; LIAO, X.; CARIN, L. Point-based policy iteration. In: **AAAI**. [S.l.: s.n.], 2007. p. 1243–1249. Citation on page 96.

KAELBLING, L. P.; LITTMAN, M. L.; CASSANDRA, A. R. Planning and acting in partially observable stochastic domains. **Artificial intelligence**, Elsevier, v. 101, n. 1-2, p. 99–134, 1998. Citations on pages 29, 36, 58, and 94.

KALAKRISHNAN, M.; PASTOR, P.; RIGHETTI, L.; SCHAAL, S. Learning objective functions for manipulation. In: IEEE. **2013 IEEE International Conference on Robotics and Automation**. [S.l.], 2013. p. 1331–1336. Citation on page 78.

KATRAKAZAS, C.; QUDDUS, M.; CHEN, W.-H.; DEKA, L. Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions. **Transportation Research Part C: Emerging Technologies**, Elsevier, v. 60, p. 416–442, 2015. Citations on pages 44 and 57.

KESTING, A.; TREIBER, M.; HELBING, D. General lane-changing model mobil for car-following models. **Transportation Research Record**, SAGE Publications Sage CA: Los Angeles, CA, v. 1999, n. 1, p. 86–94, 2007. Citation on page 85.

KUDERER, M.; GULATI, S.; BURGARD, W. Learning driving styles for autonomous vehicles from demonstration. In: IEEE. **2015 IEEE International Conference on Robotics and Automation (ICRA)**. [S.l.], 2015. p. 2641–2646. Citations on pages 76 and 95.

KURNIAWATI, H.; HSU, D.; LEE, W. S. Sarsop: Efficient point-based pomdp planning by approximating optimally reachable belief spaces. In: ZURICH, SWITZERLAND. **Robotics: Science and systems**. [S.l.], 2008. v. 2008. Citation on page 36.

KURNIAWATI, H.; YADAV, V. An online pomdp solver for uncertainty planning in dynamic environment. In: **Robotics Research**. [S.l.]: Springer, 2016. p. 611–629. Citations on pages 15, 36, and 37.

LEFÈVRE, S.; VASQUEZ, D.; LAUGIER, C. A survey on motion prediction and risk assessment for intelligent vehicles. **Robomech Journal**, Springer, v. 1, n. 1, p. 1, 2014. Citations on pages 57, 58, and 65.

LEVINE, S.; KOLTUN, V. Continuous inverse optimal control with locally optimal examples. **arXiv preprint arXiv:1206.4617**, 2012. Citation on page 76.

LEVINSON, J.; ASKELAND, J.; DOLSON, J.; THRUN, S. Traffic light mapping, localization, and state detection for autonomous vehicles. In: IEEE. **2011 IEEE International Conference on Robotics and Automation**. [S.l.], 2011. p. 5784–5791. Citation on page 44.

LITMAN, T. Autonomous vehicle implementation predictions. **Victoria Transport Policy Institute**, v. 28, 2014. Citation on page 27.

LITTMAN, M. L.; CASSANDRA, A. R.; KAELBLING, L. P. Learning policies for partially observable environments: Scaling up. In: **Machine Learning Proceedings 1995**. [S.l.]: Elsevier, 1995. p. 362–370. Citation on page 36.

LIU, W.; KIM, S. W.; PENDLETON, S.; ANG, M. H. Situation-aware decision making for autonomous driving on urban road using online POMDP. **IEEE Intelligent Vehicles Symposium, Proceedings**, IEEE, v. 2015-Augus, n. Iv, p. 1126–1133, 2015. Citations on pages 58 and 96.

MAHLER, G.; VAHIDI, A. Reducing idling at red lights based on probabilistic prediction of traffic signal timings. In: IEEE. **2012 American Control Conference (ACC)**. [S.l.], 2012. p. 6557–6562. Citation on page 45.

____. An optimal velocity-planning scheme for vehicle energy efficiency through probabilistic prediction of traffic-signal timing. **IEEE Transactions on Intelligent Transportation Systems**, IEEE, v. 15, n. 6, p. 2516–2523, 2014. Citation on page 45.

MONTEMERLO, M.; BECKER, J.; DAHLKAMP, H.; DOLGOV, D.; ETTINGER, S.; HAEHNEL, D.; HILDEN, T.; HOFFMANN, G.; JOHNSTON, D.; KLUMPP, S.; LANGER, D.; LEVANDOWSKI, A.; LEVINSON, J.; MARCIL, J.; ORENSTEIN, D.; PAEFGEN, J.; PENNY, I.; PETROVSKAYA, A.; PFLUEGER, M.; STANEK, G.; STAVENS, D.; VOGT, A.; THRUN, S. Junior: The stanford entry in the urban challenge. **Journal of field Robotics**, Wiley Online Library, v. 25, n. 9, p. 569–597, 2008. Citations on pages 28 and 57.

MOUHAGIR, H.; CHERFAOUI, V.; TALJ, R.; AIOUN, F.; GUILLEMARD, F. Using evidential occupancy grid for vehicle trajectory planning under uncertainty with tentacles. In: IEEE. **2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)**. [S.l.], 2017. p. 1–7. Citation on page 57.

NG, A. Y.; RUSSELL, S. J. *et al.* Algorithms for inverse reinforcement learning. In: **Icml**. [S.l.: s.n.], 2000. v. 1, p. 2. Citations on pages 29, 74, and 96.

OBAYASHI, M.; UTO, K.; TAKANO, G. Appropriate overtaking motion generating method using predictive control with suitable car dynamics. In: IEEE. **2016 IEEE 55th Conference on Decision and Control (CDC)**. [S.l.], 2016. p. 4992–4997. Citation on page 57.

PADEN, B.; ČÁP, M.; YONG, S. Z.; YERSHOV, D.; FRAZZOLI, E. A survey of motion planning and control techniques for self-driving urban vehicles. **IEEE Transactions on intelligent vehicles**, IEEE, v. 1, n. 1, p. 33–55, 2016. Citations on pages 27 and 74.

POMERLEAU, D. A. Alvinn: An autonomous land vehicle in a neural network. In: **Advances in neural information processing systems**. [S.l.: s.n.], 1989. p. 305–313. Citation on page 29.

RANFT, B.; STILLER, C. The role of machine vision for intelligent vehicles. **IEEE Transactions on Intelligent vehicles**, IEEE, v. 1, n. 1, p. 8–19, 2016. Citation on page 27.

ROSBACH, S.; JAMES, V.; GROSSJOHANN, S.; HOMOCEANU, S.; ROTH, S. Driving with style: Inverse reinforcement learning in general-purpose planning for automated driving. In: IEEE. **2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)**. [S.l.], 2019. p. 2658–2665. Citations on pages 76 and 78.

RUSSELL, P. N. S. **Artificial intelligence: a modern approach**. [S.l.]: Prentice hall Upper Saddle River, 2010. Citations on pages 29, 33, 35, 36, 45, 58, 74, 79, and 94.

SADIGH, D.; SASTRY, S.; SESHIA, S. A.; DRAGAN, A. D. Planning for autonomous cars that leverage effects on human actions. In: ANN ARBOR, MI, USA. **Robotics: Science and Systems**. [S.l.], 2016. v. 2, p. 1–9. Citation on page 76.

SCHWARTING, W.; ALONSO-MORA, J.; RUS, D. Planning and decision-making for autonomous vehicles. **Annual Review of Control, Robotics, and Autonomous Systems**, Annual Reviews 4139 El Camino Way, PO Box 10139, Palo Alto, California 94303-0139, USA, n. 0, 2018. Citations on pages 28 and 58.

SCHWARTING, W.; PIERSON, A.; ALONSO-MORA, J.; KARAMAN, S.; RUS, D. Social behavior for autonomous vehicles. **Proceedings of the National Academy of Sciences**, National Acad Sciences, v. 116, n. 50, p. 24972–24978, 2019. Citation on page 76.

SEZER, V. Intelligent decision making for overtaking maneuver using mixed observable markov decision process. **Journal of Intelligent Transportation Systems**, Taylor & Francis, v. 22, n. 3, p. 201–217, 2018. Citation on page 58.

SEZER, V.; BANDYOPADHYAY, T.; RUS, D.; FRAZZOLI, E.; HSU, D. Towards autonomous navigation of unsignalized intersections under uncertainty of human driver intent. In: IEEE. **2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)**. [S.l.], 2015. p. 3578–3585. Citation on page 96.

SILVA, J. A.; GRASSI, V.; WOLF, D. F. Continuous deep maximum entropy inverse reinforcement learning using online pomdp. In: IEEE. **2019 19th International Conference on Advanced Robotics (ICAR)**. [S.l.], 2019. p. 382–387. Citation on page 96.

____. Maximum entropy inverse reinforcement learning using monte carlo tree search for autonomous driving. In: **Submitted to IEEE Transactions on Intelligent Transportation Systems**. [S.l.: s.n.], 2023. Citations on pages 94, 95, and 97.

SILVER, D.; VENESS, J. Monte-carlo planning in large pomdps. **Advances in neural information processing systems**, v. 23, 2010. Citations on pages 36, 46, 56, 94, and 100.

SOMANI, A.; YE, N.; HSU, D.; LEE, W. S. Despot: Online pomdp planning with regularization. In: **Advances in neural information processing systems**. [S.l.: s.n.], 2013. p. 1772–1780. Citation on page 36.

SONG, W.; XIONG, G.; CHEN, H. Intention-aware autonomous driving decision-making in an uncontrolled intersection. **Mathematical Problems in Engineering**, Hindawi, v. 2016, 2016. Citation on page 58.

SPAAN, M. T.; VLASSIS, N. Perseus: Randomized point-based value iteration for pomdps. **Journal of artificial intelligence research**, v. 24, p. 195–220, 2005. Citation on page 96.

SUN, C.; GUANETTI, J.; BORRELLI, F.; MOURA, S. J. Optimal eco-driving control of connected and autonomous vehicles through signalized intersections. **IEEE Internet of Things Journal**, IEEE, v. 7, n. 5, p. 3759–3773, 2020. Citation on page 45.

SUN, C.; SHEN, X.; MOURA, S. Robust optimal eco-driving control with uncertain traffic signal timing. In: IEEE. **2018 Annual American Control Conference (ACC)**. [S.l.], 2018. p. 5548–5553. Citation on page 45.

SUNBERG, Z. N.; HO, C. J.; KOCHENDERFER, M. J. The value of inferring the internal state of traffic participants for autonomous freeway driving. In: IEEE. **2017 American Control Conference (ACC)**. [S.l.], 2017. p. 3004–3010. Citations on pages 58, 96, and 99.

SUNBERG, Z. N.; KOCHENDERFER, M. J. Online algorithms for pomdps with continuous state, action, and observation spaces. In: **Twenty-Eighth International Conference on Automated Planning and Scheduling**. [S.l.: s.n.], 2018. Citation on page 100.

SUTTON, R. S.; BARTO, A. G. **Reinforcement learning: An introduction**. [S.l.]: MIT press, 2018. Citation on page 45.

ŚWIECHOWSKI, M.; GODLEWSKI, K.; SAWICKI, B.; MAŃDZIUK, J. Monte carlo tree search: A review of recent modifications and applications. **Artificial Intelligence Review**, Springer, p. 1–66, 2022. Citations on pages 75 and 79.

TREIBER, M.; HENNECKE, A.; HELBING, D. Congested traffic states in empirical observations and microscopic simulations. **Physical review E**, APS, v. 62, n. 2, p. 1805, 2000. Citations on pages 76, 80, 95, and 98.

ULBRICH, S.; MAURER, M. Probabilistic online pomdp decision making for lane changes in fully automated driving. In: IEEE. **16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)**. [S.l.], 2013. p. 2063–2067. Citation on page 58.

____. Towards tactical lane change behavior planning for automated vehicles. In: IEEE. **2015 IEEE 18th International Conference on Intelligent Transportation Systems**. [S.l.], 2015. p. 989–995. Citation on page 58.

URMSON, C.; ANHALT, J.; BAGNELL, D.; BAKER, C.; BITTNER, R.; CLARK, M.; DOLAN, J.; DUGGINS, D.; GALATALI, T.; GEYER, C.; GITTLEMAN, M.; HARBAUGH, S.; HEBERT, M.; HOWARD, T. M.; KOLSKI, S.; KELLY, A.; LIKHAC, M. Autonomous driving in urban environments: Boss and the urban challenge. **Journal of Field Robotics**, Wiley Online Library, v. 25, n. 8, p. 425–466, 2008. Citations on pages 28 and 57.

WERLING, M.; ZIEGLER, J.; KAMMEL, S.; THRUN, S. Optimal trajectory generation for dynamic street scenarios in a frenet frame. In: IEEE. **2010 IEEE International Conference on Robotics and Automation**. [S.l.], 2010. p. 987–993. Citation on page 84.

WHO, W. H. O. **Global status report on road safety 2015**. [S.l.]: World Health Organization, 2015. Citation on page 27.

WU, Z.; SUN, L.; ZHAN, W.; YANG, C.; TOMIZUKA, M. Efficient sampling-based maximum entropy inverse reinforcement learning with application to autonomous driving. **IEEE Robotics and Automation Letters**, IEEE, v. 5, n. 4, p. 5355–5362, 2020. Citations on pages 76, 78, and 95.

ZHAN, W.; SUN, L.; WANG, D.; SHI, H.; CLAUSSE, A.; NAUMANN, M.; KUMMERLE, J.; KONIGSHOF, H.; STILLER, C.; FORTELLE, A. de L. *et al.* Interaction dataset: An international, adversarial and cooperative motion dataset in interactive driving scenarios with semantic maps. **arXiv preprint arXiv:1910.03088**, 2019. Citations on pages 81 and 100.

ZHOU, M.; YU, Y.; QU, X. Development of an efficient driving strategy for connected and automated vehicles at signalized intersections: A reinforcement learning approach. **IEEE Transactions on Intelligent Transportation Systems**, IEEE, v. 21, n. 1, p. 433–443, 2019. Citation on page 45.

ZIEBART, B. D.; MAAS, A. L.; BAGNELL, J. A.; DEY, A. K. *et al.* Maximum entropy inverse reinforcement learning. In: CHICAGO, IL, USA. **Aaai**. [S.l.], 2008. v. 8, p. 1433–1438. Citations on pages 40, 75, 77, 78, and 95.


ICMC
USP
SÃO CARLOS