Data-driven Behavior and Motion Planning for Autonomous Driving in Interactive Urban Environments

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard Karls Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von
M.Sc. Marcel Robert Michael Hallgarten
aus Böblingen

Tübingen
2025

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation:	24.03.2025
Dekan:	Prof. Dr. Thilo Stehle
1. Berichterstatter/-in:	Prof. Dr. rer. nat. Andreas Zell
2. Berichterstatter/-in:	Assoc. Prof.Dr.-Ing Sascha Hornauer

To everyone who has supported me throughout this journey: thank you for your encouragement and inspiration. Your presence in my life has shaped who I am today, and for that, I am deeply grateful.

Abstract

Despite the tremendous progress across nearly all vision tasks, the utopia of autonomous cars that navigate solely using vision-based sensor modalities such as camera or lidar has not yet materialized. Even though the progress in perception has led to significant advances in fundamental prerequisites such as robust detection of surrounding vehicles or pedestrians/cyclists, the rise of autonomous vehicles is held back by the conflict between high safety requirements and an infinite diversity of potential traffic scenarios.

Throughout this dissertation, we focus on challenging off-highway environments, i.e., urban traffic characterized by complex interactions among self-driving vehicles and surrounding traffic, as well as vulnerable road users. Diverse road topologies, unsignaled intersections, occlusions caused by buildings and parked vehicles, or construction zones make this particularly challenging. To efficiently navigate crowded urban environments, self-driving vehicles must understand their environment beyond accurately perceiving objects. They must focus on objects relevant to the driving task and forecast their surroundings' behavior. The result of this so-called prediction task can then be leveraged to plan a safe and efficient trajectory.

This dissertation addresses prevalent problems in the two interconnected fields of prediction and planning. We analyze how seeing them as tightly coupled tasks is paramount to reason about interactive maneuvers and demonstrate how the robustness of vehicle trajectory prediction can be improved. In addition to increased data efficiency, our methods yield more reproducible and realistic distributions over the future behavior of traffic agents. Then, we show how prediction methods can be repurposed as strong learning-based planning baselines. We also introduce a novel rule-based planning algorithm that achieves state-of-the-art performance in realistic urban traffic scenarios. Finally, we put a comprehensive set of rule-based and learning-based methods to the test in a novel benchmark centered around highly interactive maneuvers and realistic long-tail scenarios. Our results reveal that despite achieving excellent results in regular lane-following scenarios, many methods fail to generalize to these critical cases. Based on our findings, we outline promising avenues for future research that are crucial to enabling real-world autonomy.

Kurzfassung

Trotz der enormen Fortschritte im Bereich der Bildverarbeitung ist die Utopie von autonomen Fahrzeugen, die ausschließlich mit Sensormodalitäten wie Kamera oder Lidar navigieren, noch nicht Wirklichkeit geworden. Auch wenn die Fortschritte im Bereich der Umfeldwahrnehmung zu signifikanten Fortschritten bei grundlegenden Voraussetzungen wie der robusten Erkennung von umliegenden Fahrzeugen oder Fußgängern/Radfahrern geführt haben, bremst der Konflikt zwischen hohen Sicherheitsanforderungen und einer unendlichen Vielfalt möglicher Verkehrsszenarien den Aufstieg autonomer Fahrzeuge.

Der Fokus dieser Dissertation sind anspruchsvolle Off-Highway-Umgebungen, d.h. Stadtverkehr, der durch komplexe Interaktionen zwischen autonomen Fahrzeugen und anderen Verkehrsteilnehmern gekennzeichnet ist. Darüber hinaus machen eine hohe Diversität von Straßentopologien, nicht signalisierte Kreuzungen, Sichtbehinderungen durch Gebäude und parkende Fahrzeuge oder Baustellen dieses Einsatzgebiet zu einer besonderen Herausforderung. Um sich in städtischen Umgebungen effizient ans Ziel zu kommen, müssen autonome Fahrzeuge Objekte in ihrer Umgebung nicht nur wahrnehmen, sondern deren Verhalten auch verstehen und antizipieren. Das Ergebnis dieser Prädiktion kann anschließend genutzt werden, um eine sichere und effiziente Trajektorie zu planen.

Kern dieser Dissertation sind Problemstellungen in den Bereichen Prädiktion und Planung. Wir zeigen auf, dass diese beiden Bereiche als eng gekoppelte Aufgaben zu betrachten müssen, um interaktive Manöver zu verstehen. Außerdem präsentieren wir Methoden, um die Robustheit der Prädiktion von Fahrzeugtrajektorien zu verbessern. Neben einer Erhöhung der Dateneffizienz ermöglichen unsere Methoden die Vorhersage von reproduzierbaren und realistischeren Verteilungen über das zukünftige Verhalten von Verkehrsteilnehmern. Anschließend zeigen wir, wie Prädiktionsmethoden als starke, learnbasierte Planungsmodule umfunktioniert werden können. Außerdem stellen wir einen neuartigen regelbasierten Planungsalgorithmus vor, der in realistischen urbanen Verkehrsszenarien alle bisherigen Methoden übertrifft. Abschließend konzipieren wir einen umfassenden Test von regelbasierten und lernbasierten Methoden. Dazu präsentieren wir einen neuen Benchmark, der sich auf besonders interaktive Manöver und seltene Ereignisse wie Unfälle und Baustellen konzentriert. Nach unserer Erkenntnis erzielen viele Methoden zwar hervorragende Ergebnisse in regulären Szenarien, die größtenteils durch einfaches Folgen der Fahrspur gelöst werden können, sie sind jedoch nicht in der Lage, in diesen kritischen Fällen sicher und zielorientiert zu fahren. Auf der Grundlage unserer Ergebnisse zeigen wir vielversprechende Richtungen für zukünftige Forschung auf, die für die Ermöglichung von autonomem Fahren in der realen Welt von entscheidender Bedeutung sind.

Contents

Chapter 1

Introduction

Individual transportation once changed how we live, work, and meet by making even larger distances conveniently traversable. In the same way, autonomous driving can mark a new era by making transportation safer, more efficient, and more accessible. This era will be characterized by fewer accidents and thus fewer traffic fatalities, unimpeded access to mobility for the elderly or disabled, more reliable and efficient supply chains, as well as new business fields centered around on-demand mobility and the technology enabling it. However, this long-established utopia has not yet materialized despite these efforts lasting almost a century (Wetmore, 2003). High safety requirements and an infinite diversity of potential traffic situations make this interdisciplinary task a challenging problem.

To successfully navigate diverse traffic conditions, a driver (human or robot) has to perceive and understand the environment to make informed decisions, i.e., plan a safe and goal-directed behavior and ultimately control the vehicle according to the plan. Perceiving the environment involves recognizing infrastructure (e.g., lane markings, traffic lights, and signs) and surrounding objects (e.g., vehicles and other traffic participants such as cyclists and pedestrians). Moreover, a driver needs to understand which objects are relevant. This can refer, for example, to identifying the traffic light that corresponds to the current lane or to identifying the leading vehicle. To make a socially acceptable plan based on the perceived environmental information, drivers need to anticipate the intentions of other vehicles and understand the effects of their own actions. For instance, starting an unprotected left turn is dangerous if it requires the oncoming traffic to brake hard, but it might be reasonable if the oncoming vehicle is only required to decelerate slightly or if it brakes already to allow for a safe turn. Finally, the core of the driving task is to control the vehicle's steering and acceleration to follow the plan while always staying ready to react quickly to unforeseen events, such as a pedestrian stepping onto the street. These aspects are already challenging by themselves, especially given that they must be performed simultaneously and under real-time constraints. Moreover, robust generalization is required to handle novel object appearances such as new car models, varying weather conditions, or previously unseen events (e.g., ice falling from a truck).

Still, humans can acquire the relevant skills in less than 50 hours. The observation that tasks that seem to be easy for human beings tend to pose major challenges for robots and vice versa was already described in the 1980s, commonly known as Moravecs

Paradox: "It is comparatively easy to make computers exhibit adult level performance on intelligence tests or playing checkers, and difficult or impossible to give them the skills of a one-year-old when it comes to perception and mobility" (Moravec, 1988)

A prevalent strategy to approach this complex task is to split it into several smaller subproblems, namely *perception*, *prediction*, *planning*, and *control*. Perception creates an environmental model from sensor data and prior knowledge (such as map information). Thus, it involves localization, mapping, object detection, and tracking. Next, prediction is centered around forecasting the environment to make informed decisions in the planning step, which generates a spatiotemporal trajectory describing the desired states of the ego-vehicle at future timesteps. Subsequently, a downstream controller infers optimal acceleration and steering commands to follow this trajectory. While this divide-and-conquer strategy eases the individual development of subsystems, it comes at the cost of the following pitfalls: First, improving the performance of one subsystem often does not generalize to better performance in the overall driving task. For instance, an object detector that improves significantly in bounding-box estimation for far-away pedestrians at the cost of missing some nearby pedestrians entirely might achieve better results with respect to task-specific metrics. However, it will likely result in more safety-critical planning decisions and, thus, worse driving performance. Second, this often results in slow iteration cycles, as failures of the system that are patched with updates to one subsystem entail a re-validation of the entire system. Last, defining and engineering interfaces is non-trivial as they must suit all corner cases, such as object classes that did not exist at development time (e.g., e-scooters or autonomous shuttles). Systems that are end-to-end differentiable across several subtasks aim to overcome this by optimizing the entire system, including the interfaces, with respect to the final driving performance. Such systems can span from perception to control, directly mapping sensor data to steering and acceleration commands, or from perception to planning, mapping sensor data to a trajectory, which is then tracked by a vehicle-specific traditional controller. They can take the form of modular systems consisting of several differentiable subsystems with interpretable intermediate output representations (Karkus *et al.*, 2023; Li *et al.*, 2023a) or monolithic black-box models where the subtasks are no longer separated and no introspection is possible.

This work's focus is autonomous driving in challenging interactive urban scenarios. The urban environment demands robust generalization to diverse road topologies and highly multimodal future scenarios. For instance, a vehicle observed at a four-way intersection could stop, turn left or right, and continue straight. Moreover, the driver could park the vehicle on the roadside or make a U-turn. Moreover, urban traffic involves several highly interactive scenarios, such as unprotected turns, lane changes or merges in dense traffic, and situations where vehicles must navigate safely near vulnerable road users such as pedestrians and cyclists. Such interaction is only possible if the driver understands how the surrounding traffic will behave, how to react appropriately, and how his own actions impact this behavior. On a system level, it requires interconnected reasoning about surrounding traffic's future intention, i.e., *prediction* and decision-making for the ego-vehicle, i.e., *planning*.

In particular, this work addresses the following problems to improve the state-of-the-art in the field of urban autonomous driving:

- **System-design**: Integrating prediction and planning in a way that models the mutual influence of the ego-vehicle and the surrounding traffic is an open challenge. In a survey, this work assesses design paradigms and analyzes implications on safety and interactive behavior.
- **Robustness** of learning-based vehicle trajectory prediction with respect to generalization to novel map topologies and with respect to the reproducibility of the distribution of future behaviors represented by samples from a latent distribution. Both are critical safety requirements and fundamental pillars for real-world deployment.
- **Goal-conditioning** is a key difference between vehicle trajectory prediction and learning-based planning, as a planner has to use navigation information to progress toward the goal. This work shows, how goal-conditioning can be achieved for graph-based models at inference time so that expressive prediction models can be repurposed for planning.
- **Rule-based planning** is an alternative to learning-based planning. This work assesses the performance of both paradigms in an urban driving benchmark and presents a novel state-of-the-art rule-based planner.
- **Generalization** to long-tail and interactive scenarios is paramount to autonomy. Thus, this work presents a novel benchmark centered around such challenging scenarios and assesses the capabilities of state-of-the-art methods.

These contributions to the field of prediction and planning rely on the assumption of perfect perception. This enables efficient and realistic evaluation without cumbersome sensor simulation. However, as shown in (Karkus *et al.*, 2023), differentiable modular prediction and planning systems can be used to build end-to-end differentiable pipelines that propagate gradients from the final planning task to the upstream perception module.

1.1 Contribution Outline

This work is based on the following publications:

1. Marcel Hallgarten, Martin Stoll, and Andreas Zell. “From Prediction to Planning with Goal Conditioned Lane Graph Traversals”. 2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC). IEEE, 2023.

   In this work, we propose to repurpose trajectory prediction models trained with supervised learning on driving logs recorded with a human expert for planning. In this regard, conditioning the forecast on an existing navigation goal plays a key role. We show that trajectory prediction models that distinguish between behavior and motion level open an opportunity to leverage a novel method of goal-conditioning. We apply this method to a state-of-the-art trajectory prediction model and successfully evaluate it in a realistic closed-loop simulation.

2. Faris Janjoš, Marcel Hallgarten, Anthony Knittel, Maxim Dolgov, Andreas Zell, and J. Marius Zöllner. “Conditional Unscented Autoencoders for Trajectory Prediction”. Accepted at European Conference on Computer Vision (ECCV) Workshop “ROAD++”, 2024.

   In this work, we apply a conditional variational autoencoder to the task of vehicle trajectory prediction. We demonstrate that common simplistic latent-spaces are insufficient to outperform traditional encoder-decoder models. Thus, we propose to leverage a more expressive and more structured Gaussian mixture latent space. Moreover, we show how this opens a new oportunity to estimate a parametric distribution in output space, namely by applying the unscented transform, which is well known from control theory, to project the latent distribution to the output space. We apply this method to the INTERACTION prediction dataset and showcase a significant improvement over traditional feedforward encoder-decoder methods.

3. Marcel Hallgarten, Ismail Kisa, Martin Stoll, and Andreas Zell. “Stay on Track: A Frenet Wrapper to Overcome Off-road Trajectories in Vehicle Motion Prediction”. 2024 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2024.

   In this work, we propose a framework that represents the scene context for trajectory prediction models in a path-relative Frenet frame conditioned on a lane centerline. Our framework can be applied to many state-of-the-art methods without changing the model architecture. We show that this reduces off-road predictions in difficult scenarios by 90% with only a slight reduction in prediction accuracy in regular cases. Moreover, we explore different methods to aggregate prediction results conditioned on different lanes and test our method with two state-of-the-art trajectory prediction methods.

4. Daniel Dauner, Marcel Hallgarten, Andreas Geiger, and Kashyap Chitta. "Parting with Misconceptions about Learning-based Vehicle Motion Planning". Conference on Robot Learning (CoRL). PMLR, 2023.

   In this work, we assess the state of vehicle motion planning in challenging urban environments. We present two simple and efficient yet very effective baselines: A rule-based reactive driving policy that achieves state-of-the-art results in closed-loop driving and a competitive learned MLP-based open-loop model. Combining both models resulted in winning the nuPlan Planning Challenge 2023.

5. Steffen Hagedorn*, Marcel Hallgarten*, Martin Stoll, and Alexandru Condurache. "The Integration of Prediction and Planning in Deep Learning Automated Driving Systems: A Review". Transactions on Intelligent Vehicles (T-IV). IEEE, 2024.

   In this work, we review architectures, interfaces and design paradigms in the state-of-the-art of vehicle trajectory prediction and planning methods. We assess relevant literature from the perspective of combining both sub-tasks into an integrated autonomous driving system. We focus on the implications that design choices have on safety and the interactive behavior of the overall system. Finally, we analyze trends and challenges and outline promising directions for future research.

6. Marcel Hallgarten, Ismail Kisa, Martin Stoll, Katrin Renz, and Andreas Zell. "Can Vehicle Motion Planning Generalize to Realistic Long-Tail Scenarios?". International Conference on Intelligent Robots and Systems (IROS). IEEE/RSJ, 2024

   In this work, we propose a novel benchmark to test the generalization capabilities of motion planning methods, named interPlan. Our benchmark augments real-world scenarios from the nuPlan dataset and comprises challenging, rare scenarios such as encountering jaywalkers or construction zones. Moreover, it entails highly interactive driving scenarios, such as lane changes in dense traffic with diverse traffic agent policies ranging from conservative to assertive driving. We analyze a comprehensive set of state-of-the-art motion planning methods, including rule-based and learning-based approaches, and reveal critical shortcomings in their generalization capabilities. Moreover, we provide two methods based on large language models as strong baselines for our benchmark.

In addition, the following work was done during the doctorate, but it will not be covered in this dissertation.

1. Daniel Dauner, Marcel Hallgarten, Tianyu Li, Xinshuo Weng, Zhiyu Huang, Zetong Yang, Hongyang Li, Igor Gilitschenski, Boris Ivanovic, Marco Pavone, Andreas Geiger, and Kashyap Chitta. "NAVSIM: Data-Driven Non-Reactive Autonomous Vehicle Simulation and Benchmarking". Advances in Neural Information Processing Systems (NeurIPS), 2024.

An overview of the contribtutions made to each publication can be found in the appendix.

Moreover, the following workshops and competitions were co-organized as part of the doctorate:

1. Sascha Hornauer, Maximilian Naumann, Marcel Hallgarten, Eike Rehder, Jiachen Li, Wei Zhan, Martin Lauer, Masayoshi Tomizuka, Arnaud de La Fortelle, Christoph Stiller. "Interaction-driven Behavior Prediction and Planning for Autonomous Vehicles", Intelligent Vehicles Symposium, 2024

2. Kashyap Chitta, Daniel Dauner, Marcel Hallgarten, Boris Ivanovic, Marco Pavone, Igor Gilitschenski. "Autonomous Grand Challenge. Track: End-to-End Driving at Scale", Conference on Computer Vision and Pattern Recognition, 2024

3. Marcel Hallgarten, Martin Stoll, Faris Janjos, Felicia Ruppel, Andreas Zell, Abhinav Valada, Marco Pavone, Igor Gilitschenski. "Workshop on Interaction-aware Autonomous Systems", International Conference on Intelligent Robots and Systems, 2024.

4. Maytheewat Aramrattana, Nassim Belmecheri, Atia Cortés, Arnaud de La Fortelle, Frank Diermeyer, Marcel Hallgarten, Sascha Hornauer, Jonas Jansson, Martin Lauer, Jiachen Li, Maximilian Naumann, Michael Oehl, Karla Quintero, Eike Rehder, Andreas Schrank, Christoph Stiller, Masayoshi Tomizuka, Marek Vanzura, Pavan Vasishta, Maria Wolf, Yanbin Wu, Wei Zhan. "Advancing Automated Driving in Highly Interactive Scenarios through Behavior Prediction, Trustworthy AI, and Remote Operations", Intelligent Vehicles Symposium, 2025, **(Ongoing)**

5. Marcel Hallgarten, Kashyap Chitta, Daniel Danuer, Tianyu Li, Wei Cao. "Autonomous Grand Challenge 2025. Track: NAVSIM End-to-End Driving Challenge", Conference on Computer Vision and Pattern Recognition, 2025, **(Ongoing)**

1.2 Structure of the Document

The remainder of the thesis is structured as follows:

Chapter 2 We start by laying out the fundamentals of the autonomous driving task, especially focusing on the subtasks of vehicle trajectory prediction and motion planning. Thus, we present our paper "The Integration of Prediction and Planning in Deep Learning Automated Driving Systems: A Review" (Hagedorn *et al.*, 2023), which reviews the state of the field in these two tasks and describes architectures, interfaces, and design choices.

Chapter 3 Next, we present our works in vehicle trajectory prediction. Our paper "Conditional Unscented Autoencoders for Trajectory Prediction" (Janjoš *et al.*, 2023b) presents a novel model architecture based on conditional variational autoencoders which introduces several new components, including a deterministic sampling procedure and a more structured Gaussian Mixture Model (GMM) latent space. Putting both together results in a robust, deterministic, and accurate parametric estimation of the output distribution.

Chapter 4 Subsequently, our work "Stay on Track: A Frenet Wrapper to Overcome Off-road Trajectories in Vehicle Motion Prediction" (Hallgarten *et al.*, 2023b) addresses the prevalent problem of off-road predictions, i.e., trajectory forecasts that do not comply with map information and thus waste prediction budget on inadmissible forecasts. Our paper overcomes this by training prediction models in a framework that involves a Frenet frame wrapper. This wrapper transforms the scene representation, fed into the model into a path-relative frame conditioned on a lane centerline. As the model output is defined in the same frame, this creates a strong inductive bias towards lane-following, even when facing difficult road topologies. It thus reduces offroad predictions significantly.

Chapter 5 Subsequently, we show how trajectory prediction models can be repurposed for the planning task, serving as easy-to-obtain and effective baselines. A core difference between the prediction of surrounding vehicles and the planning for the ego-vehicle is that in the latter case, the trajectory has to be conditioned on a navigation goal, commonly provided by a GPS navigation system. Therefore, we present our paper "From Prediction to Planning with Goal Conditioned Lane Graph Traversals" (Hallgarten *et al.*, 2023a).

Chapter 6 We then benchmark this method against rule-based planning methods. Presenting our paper "Parting with Misconceptions about Learning-based Vehicle Motion Planning" (Dauner *et al.*, 2023), we introduce the Predictive Driver Model (PDM), a rule-based planning model that outperforms all prior learning-based and rule-based methods in the closed-loop nuPlan driving benchmark.

Chapter 7 Fueled by the impressive results of the PDM methods presented in the previous chapter, we ask the question if these capable methods are able to generalize to realistic and challenging but rarely occurring driving scenarios - an aspect that is crucial for real-world deployment. Therefore, we assess the state of the field in a novel benchmark presented in our paper "Can Vehicle Motion Planning Generalize to Realistic Long-tail Scenarios?" (Hallgarten *et al.*, 2024). It is centered around such long-tail scenarios as well as highly interactive driving situations. Our benchmark reveals critical shortcomings in the state-of-the-art and points to critical aspects for future research.

Chapter 8 Finally, we draw a short conclusion on the work presented in this thesis and provide an brief outlook.

Chapter 2

Foundations

2.1 Introduction

Automated Driving (AD) remains a challenging endeavour. It is usually split into the subtasks of perception, prediction, planning, and control (Yurtsever *et al.*, 2020; Gruyer *et al.*, 2017; Wen and Jo, 2022). Perception processes sensor inputs to create a model of the environment. Prediction and planning build upon this model and make future motion forecasts for surrounding traffic agents and a plan for the controlled ego vehicle. Traditional, modular systems (cf. Fig. 2.2) address prediction and planning as separate tasks. The predicted behavior of surrounding traffic agents is leveraged to plan a suitable behavior for the ego vehicle. However, this sequential ordering is inherently reactive and cannot represent bidirectional interaction between the ego vehicle and other traffic agents (Sun *et al.*, 2022). In fact, prediction and planning are no sequential problems and should be tightly coupled in automated driving systems (Rhinehart *et al.*, 2021; Ngiam *et al.*, 2022; Huang *et al.*, 2022b). Fig. 2.1 highlights the interdependence of both tasks.

2.1.1 Scope

In this chapter, we review methods to integrate prediction and planning into an automated driving system. Prediction is the task of anticipating intents and future trajectories of observed traffic agents (Lee *et al.*, 2017), and planning is about finding the best possible trajectory w.r.t. previously defined criteria for a controlled vehicle (Pomerleau, 1988). Deep Learning (DL)-based methods follow a data-centric approach to tackle these problems (Wang *et al.*, 2019a) and have led to significant improvements in many fields (Dong *et al.*, 2021). Works in prediction and planning increasingly employ DL-based approaches as well (Grigorescu *et al.*, 2020; Kuutti *et al.*, 2020; Huang and Chen, 2020; Huang *et al.*, 2022a) and represent an alternative to traditional, rule-based methods (Treiber *et al.*, 2000; Kesting *et al.*, 2007; Shalev-Shwartz *et al.*, 2017) and constrained optimization (Liu *et al.*, 2017). In this chapter, we focus on DL-based methods.

An overview of different architectures of automated driving systems is shown in Fig. 2.2. It can take three forms ranging from modular systems over end-to-end (E2E) differentiable modular systems to monolithic E2E systems. Modular and modular E2E

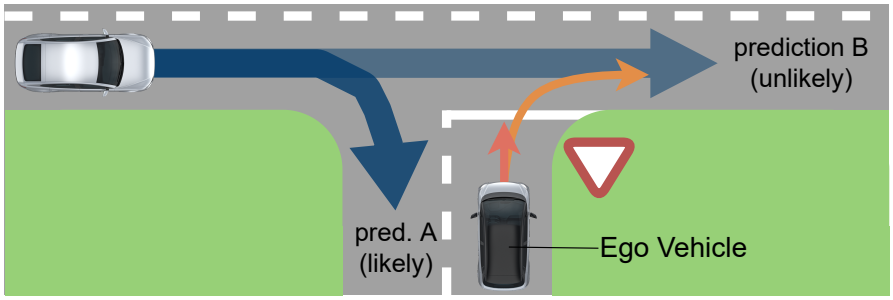


Figure 2.1: **How should the ego vehicle behave?** Various behaviors are possible for the ego vehicle. Here, we depict two in red and orange. Their consequences depend on the behavior of the observed surrounding vehicle, which is depicted by blue arrows (prediction A or prediction B). At the same time, the surrounding vehicle might react to the ego vehicle's action, i.e. the surrounding vehicle's behavior also depends on the ego vehicle's decision. Different methods exist to forecast the behavior of the surrounding vehicle, and various ways exist to leverage this to decide on a safe and goal-directed plan for the ego-vehicle. In this survey, we systematically categorize and review methods that integrate prediction and planning for self-driving vehicles. We highlight their capabilities and limitations and discuss prospects for future research.

systems consist of clearly defined components for different tasks (Hu *et al.*, 2022; Müller *et al.*, 2018; Hawke *et al.*, 2020; Scheel *et al.*, 2022; Li *et al.*, 2023a). Traditionally, they are arranged sequentially. We will discuss these systems in Sec. 2.3. In contrast, monolithic E2E methods employ a single neural network for perception, prediction, and planning so that depending on the architecture, no boundaries between the modules can be drawn (Pomerleau, 1988; Bojarski *et al.*, 2016; Chitta *et al.*, 2022). As the order of the subtasks cannot be determined, we name this integration of prediction and planning "undirected" (c.f. Sec. 2.4). In this chapter, we emphasize the shortcomings of sequential and undirected approaches and highlight works that integrate prediction and planning in a more sophisticated way to enable bidirectional interaction of self-driving vehicles and surrounding traffic (c.f. Sec. 2.5). In the following, we refer to the component of an automated driving system that integrates prediction and planning as an Integrated Prediction and Planning System (IPPS).

2.1.2 Contribution and Outline

By discussing automated driving systems with a focus on the integration of prediction and planning, we take a novel perspective. IPPSs can be classified into three categories as depicted in Fig. 2.2. Accordingly, this survey is structured into three main chapters: sequential (Sec. 2.3), undirected (Sec. 2.4), and bidirectional IPPSs (Sec. 2.5). While surveys on classical methods (Lefèvre *et al.*, 2014; Claussmann *et al.*, 2019; Huang and Chen, 2020; Leon and Gavrilescu, 2021), stand-alone DL-based prediction (Mozaffari *et al.*,

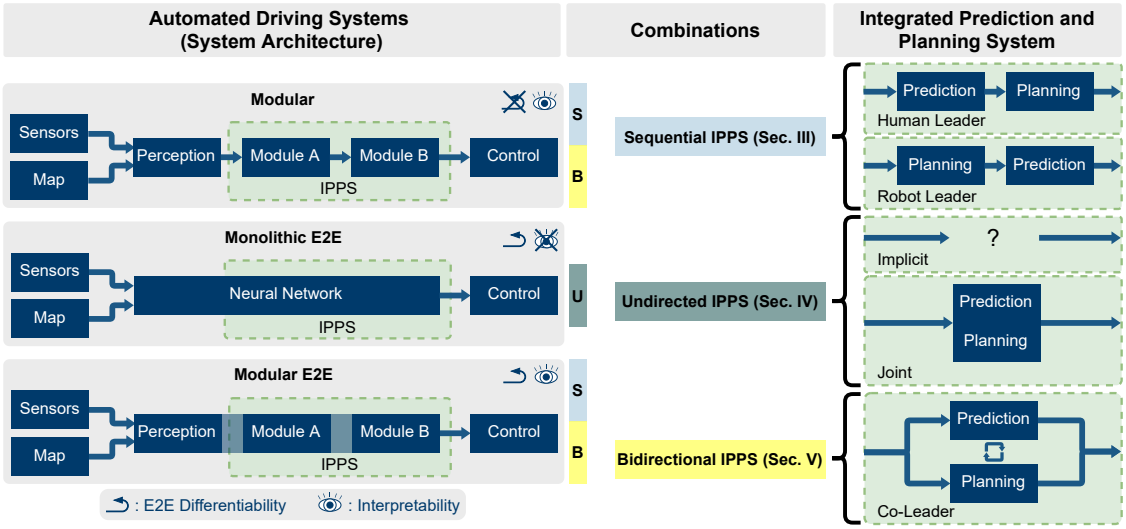


Figure 2.2: Overview of automated driving systems. There are three system architectures shown on the left. Modular systems consist of individual modules, whose interfaces provide interpretability but restrict information flow and end-to-end differentiability. In contrast, monolithic E2E systems are end-to-end differentiable but not interpretable. Modular E2E systems combine both properties: they are end-to-end differentiable and interpretable. All three system architectures comprise an integrated prediction and planning system (IPPS), depicted in green. Our work focuses on this very part and identifies three paradigms to integrate prediction and planning, as shown on the right. Sequential IPPSs condition one task on the other. Undirected IPPSs allow for more complex interactions but provide low interpretability. Bidirectional IPPSs explicitly ensure that both tasks are mutually conditioned on each other. Our review analyzes these integration paradigms and highlights their compatibility with different system architectures.

2020; Liu *et al.*, 2021b; Ding and Zhao, 2023; Huang *et al.*, 2022a) or planning (Schwarting *et al.*, 2018), and E2E AD (Chen *et al.*, 2023b) already exist, we observe that recent methods thoughtfully design the interplay of prediction and planning (Huang *et al.*, 2023b; Jiang *et al.*, 2023a; Hu *et al.*, 2023c; Huang *et al.*, 2023a). To substantiate this observation with broad evidence and a theoretical foundation, we summarize our contributions as follows:

- We propose a categorization for the integration of prediction and planning based on the dependencies between both tasks (cf. Sec. 2.3, 2.4 and 2.5). We investigate how these categories relate to system architectures, behavioral aspects, and safety.
- In particular, we provide a comprehensive overview of design choices in prediction and planning modules of modular (E2E) architectures (cf. Sec. 2.3.1 and 2.3.2) and discuss their impact on interaction and system-level behavior.
- We reveal gaps in state-of-the-art research and point out promising directions of future research based on the identified categorization (Sec. 2.6).

2.2 Task Definitions

In the following, we start by introducing the terminology and notation for the relevant task definitions. We adopt a similar terminology to that proposed by (Mozaffari *et al.*, 2020) and partition the actors in a traffic scenario into the self-driving ego vehicle (EV) and the surrounding vehicles (SVs). The EV is equipped with sensors that provide information on the environment. The state history of vehicle i over the time interval $t - t_{\text{obs}}, \dots, t$ is

$$X_v = \{s^{(v)}_{t_0 - t_\text{h}}, \dots, s^{(v)}_{t_0-1}, s^{(v)}_{t_0}\}. \tag{2.1}$$

Each state s comprises 2D or 3D positional information and further optional information like heading angle, speed, static attributes, or goal information in the case of the EV. Hence, X_{EV} denotes the EV's past states. Similarly,

$$\overline{\mathbf{X}}_{\text{SV}} = \{X_1, X_2, \dots, X_m\} \tag{2.2}$$

refers to the past states of all SVs. Analogously, the future states of vehicle i within a prediction horizon of t_{pred} are

$$Y_v = \{s^{(v)}_{t_0+1}, s^{(v)}_{t_0+2}, \dots, s^{(v)}_{t_0+t_\text{f}}\}. \tag{2.3}$$

and the future states of all SVs are $\overline{\mathbf{Y}}_{\text{SV}}$. In the following, state sequences are also referred to as trajectories. Additional scene information, such as a semantic map or traffic signs and traffic light states are represented by I.

Following (Lee *et al.*, 2017), we state trajectory *prediction* as the task of estimating a probability distribution

$$P_{\text{pred}} = P(\overline{\mathbf{Y}}|\overline{\mathbf{X}}, I) \tag{2.4}$$

that maps state histories of m observed vehicles in $\overline{\mathbf{X}}$ to the future trajectories of n predicted vehicles in $\overline{\mathbf{Y}}$. The distribution P_{pred} accounts for inherent uncertainties in the forecasting task and is often modeled by a discrete set of samples with corresponding probabilities (Varadarajan *et al.*, 2022; Deo and Trivedi, 2020; Salzmann *et al.*, 2020; Liang *et al.*, 2020). Some methods omit scene information I and infer trajectories based on a state history alone (Mercat *et al.*, 2020; Xu *et al.*, 2022b; Schmidt *et al.*, 2022). Commonly $\overline{\mathbf{X}}$ includes all vehicles in the scene, i.e. X_{EV} and $\overline{\mathbf{X}}_{\text{SV}}$. Depending on the vehicles in $\overline{\mathbf{Y}}$, different variants of prediction can be formulated: single-agent prediction models the future trajectory for each SV individually (Bhattacharyya *et al.*, 2024). However, estimating a joint prediction $\overline{\mathbf{Y}}_{\text{SV}}$ from a set of n single-agent predictions Y_{SV} is not trivial since the number of possible combinations grows exponentially with the number of agents n. Not all combinations are meaningful, and finding realistic ones with heuristics and joint optimization is cumbersome (Gilles *et al.*, 2021b). To avoid this, joint prediction directly estimates a joint distribution for multiple SVs (Casas *et al.*, 2020a).

In *planning*, the task is to find a single suitable trajectory for the EV that can be passed on to a downstream motion controller. Hence, we define planning to be a function f that maps the observational inputs $\overline{\mathbf{X}}_{\text{SV}}$, and X_{EV} as well as the context information I to a future trajectory Y_{EV}, i.e.

$$Y_{\text{EV}} = f(X_{\text{EV}}, \overline{\mathbf{X}}_{\text{SV}}, I). \tag{2.5}$$

In many cases, the function f also uses the prediction $\overline{\mathbf{Y}}_{\text{SV}}$, i.e.

$$Y_{\text{EV}} = f(X_{\text{EV}}, \overline{\mathbf{X}}_{\text{SV}}, I, \overline{\mathbf{Y}}_{\text{SV}}). \tag{2.6}$$

The definitions show that planning can be considered a special case of single-agent prediction where the output distribution models only a single trajectory. However, in contrast to prediction, the planning trajectory must be conditioned on a navigation goal. Moreover, it has to be stable in closed-loop, i.e. it must be kinematically feasible and result in safe and efficient driving behavior when fed to a downstream controller (Rajamani, 2011).

2.3 Sequential IPPS

To date, the sequential integration of prediction and planning is the prevalent and most widely adopted integration principle. Sequential IPPSs execute prediction and planning as separate tasks. Prior knowledge is used to design the interplay of the modules. Depending on the interfaces, such systems can still be end-to-end differentiable. Accordingly, sequential integration can be found in modular and modular E2E systems (c.f. Fig. 2.2). To

investigate different sequential system designs in more depth, we start by taking a closer look at the individual components of prediction and planning and discuss their impact on interaction and system-level behavior.

2.3.1 Prediction

Input Representations

In the context of IPPSs for automated vehicles, agent states $\overline{\mathbf{X}}$ and map I are the most important information to represent. Two major representations exist in DL-based approaches: Rasterized and sparse.

Rasterized representations use dense, fixed-resolution grid structures that often have multiple channels (Casas *et al.*, 2018; Park *et al.*, 2020; Chen *et al.*, 2022; Kamenev *et al.*, 2022; Stoll *et al.*, 2023; Janjoš *et al.*, 2021). Each channel encodes different information on agent states. They are commonly used in combination with rasterized high-definition maps (HD maps) (Bansal *et al.*, 2018; Mi *et al.*, 2021) and represented in a Bird's Eye View (BEV) (Philion and Fidler, 2020; Li *et al.*, 2022b). The BEV allows to fuse multiple EV sensor inputs and different sensing modalities into a shared and interpretable representation (Lee *et al.*, 2017). It further establishes a common coordinate system for all vehicles, facilitating to model interactions within the scene (Djuric *et al.*, 2018). Dense grids are well-suited for processing with powerful Convolutional Neural Network (CNN) architectures (LeCun *et al.*, 1989; Zeng *et al.*, 2019; Rhinehart *et al.*, 2018a; Wang *et al.*, 2019a; Chen *et al.*, 2019; Codevilla *et al.*, 2019; Rhinehart *et al.*, 2019; Sadat *et al.*, 2020; Chen *et al.*, 2020; Zeng *et al.*, 2020; Cui *et al.*, 2021; Casas *et al.*, 2021; Konev *et al.*, 2022). However, embedding all observations in a grid structure leads to information loss due to quantization errors (Diehl *et al.*, 2019). Moreover, the locally restricted receptive field and the limited resolution of CNNs can hamper interaction modeling.

Sparse representations use vectors to describe all objects in a scene. The per-object vectors are then either encoded into a latent representation jointly, e.g., with a Transformer, or individually and then aggregated, e.g., with Graph Neural Network (GNN) (Kipf and Welling, 2016). Objects are represented by polygons, or point sets (Ngiam *et al.*, 2022; Gao *et al.*, 2020; Liang *et al.*, 2020; Zhao *et al.*, 2021b; Liu *et al.*, 2021c; Varadarajan *et al.*, 2022; Mu *et al.*, 2024; Deo *et al.*, 2022; Janjoš *et al.*, 2022a; Deo and Trivedi, 2020; Mo *et al.*, 2022; Pittner *et al.*, 2024). The map is commonly described by a set of lanes represented by polylines and an adjacency matrix. Objects and map elements are then encoded into fixed-size latent features, for instance, via Multilayer Perceptron (MLP) (Rosenblatt, 1958) or Recurrent Neural Networks (RNN) (Rumelhart *et al.*, 1986). To encode sequential signals like state histories $\overline{\mathbf{X}}$, 1D CNNs (Kim *et al.*, 2021), RNNs like LSTMs (Hochreiter and Schmidhuber, 1997) and GRUs (Chung *et al.*, 2014), and Transformers (Vaswani *et al.*, 2017) are applied (Liang *et al.*, 2020; Gilles *et al.*, 2022; Liu *et al.*, 2021c). Some works like Trajectron++ (Salzmann *et al.*, 2020) and MFP (Tang and

Salakhutdinov, 2019) combine rasterized HD maps with sparse object representations.

Throughout the past years, a clear trend towards sparse input representations can be observed in prediction modules (Zhou *et al.*, 2022; Nayakanti *et al.*, 2022; Shi *et al.*, 2022a; Cui *et al.*, 2022; Jiang *et al.*, 2023b; Liu *et al.*, 2024; Philion *et al.*, 2024; Liao *et al.*, 2024). In the context of integrating prediction and planning, an advantage of sparse representations is that they are object-based. Interactions between the EV and SVs can therefore be modeled in a targeted manner. This improves the explainability of a system and provides a clear understanding of how surrounding vehicles influence the ego-plan.

Interaction Modelling

Provided with an accurate scene representation, interaction modeling between perceived objects is a decisive factor for the successful integration of prediction and planning (Sun *et al.*, 2022; Tolstaya *et al.*, 2021). Modeling interactions between vehicles helps to understand how the actions of a vehicle affect the behavior of other traffic participants (Liao *et al.*, 2024). This model is subject to various influences, including traffic rules, physics, and common sense reasoning. Interaction modeling also includes interactions between map elements and vehicles. For example, modeling the relation of vehicles to static map elements like lane markings, traffic signs, etc. is important to understand feasible corridors in which trajectories could be localized. Some models employ rule-based heuristics to guide interaction modeling, e.g., ScePT (Chen *et al.*, 2022) only models interaction among nearby vehicles. Similarly, heterogeneous graph attention network (HGAT) (Demmler *et al.*, 2024) and graph-based interaction-aware trajectory prediction (GRIP) (Li *et al.*, 2019b) use heuristics to identify which pairs of nodes in the scene graph are to be connected with edges. Other methods rely entirely on neural components to learn these relations. Altogether, an interaction model should yield an understanding of how the scene can potentially evolve in the future.

Recapitulating recent methods exposes that diverse architectures are used to realize interaction modeling. *RNNs* are only used in early prediction models like DESIRE (Lee *et al.*, 2017) or Trajectron++ (Salzmann *et al.*, 2020) and combined with an aggregation operator like spatial pooling or attention. Alternatively, *CNN* methods apply 2D convolutions to implicitly capture interaction within the kernel size (Luo *et al.*, 2018; Cui *et al.*, 2019; Chai *et al.*, 2019). Compared to sequence processing-based approaches, higher importance is assigned to spatial interaction. However, no explicit interactions between designated objects are modeled. This shortcoming is addressed by *GNNs and Graph-Attention*, which explicitly model interactions between individual agents. They either combine multiple agents' features (Zhang *et al.*, 2024) and process them with graph convolution operators (Wang *et al.*, 2019c; Kipf and Welling, 2016; Li *et al.*, 2019b,a) or perform graph attention (Bahdanau *et al.*, 2014; Luong *et al.*, 2015; Velickovic *et al.*, 2017; Pan *et al.*, 2020; Gu *et al.*, 2021) to aggregate information. Recently, *Transformers* (Vaswani *et al.*, 2017) are widely adopted for interaction modeling due to their global receptive field and attention mechanism (Li *et al.*, 2020; Liu *et al.*, 2021c; Yuan *et al.*,

2021; Ngiam *et al.*, 2022; Zhou *et al.*, 2022; Nayakanti *et al.*, 2022; Shi *et al.*, 2022a; Jiang *et al.*, 2023b; Philion *et al.*, 2024; Chen *et al.*, 2021; Quintanar *et al.*, 2021; Girgis *et al.*, 2021; Postnikov *et al.*, 2021; Singh and Srivastava, 2022; Hazard *et al.*, 2022; Zhang *et al.*, 2022b; Wonsak *et al.*, 2022; Amirloo *et al.*, 2022; Hu *et al.*, 2023a). All SVs can be predicted simultaneously (Liu *et al.*, 2021c), and the impact of vehicles on each others' behaviors across different timesteps can be modeled by masking (Yuan *et al.*, 2021). Attention is applicable along the spatial and temporal dimension jointly (Yuan *et al.*, 2021), or individually (Chen *et al.*, 2023c). The latter break down into sequential (Arnab *et al.*, 2021; Ho *et al.*, 2019) and interleaved (Ngiam *et al.*, 2022; Nayakanti *et al.*, 2022) attention. Experiments with Wayformer (Nayakanti *et al.*, 2022) and Scene-Transformer (Ngiam *et al.*, 2022) show that joint and interleaved processing outperform sequential approaches. As joint processing causes a higher computational burden, the interleaved paradigm balances best between performance and computational demand. This property of Transformers can provide explainability to the integration of prediction and planning. Attention can be utilized flexibly to achieve locality restrictions, causality, and an explicit interaction design. For example, a sequential IPPS could first model interactions between SVs and then include the EV for planning.

Coordinate Frame

Irrespective of the scene representation itself, the coordinate frame is a fundamental design choice that affects interaction modeling and thus the integration of prediction and planning. Applying a global coordinate system with a fixed viewpoint for the whole scene (Khandelwal *et al.*, 2020; Gilles *et al.*, 2021b; Zeng *et al.*, 2021; Ngiam *et al.*, 2022) is computationally efficient as it can be shared across both modules. However, the frame is not viewpoint-invariant, i.e., the predictions and the plan change if the origin of the coordinate system is moved within the scene. This leads to low sample efficiency and impairs generalization (Reiter *et al.*, 2024). Therefore, the origin is commonly fixed to the center of the EV. While this normalization with respect to the EV is a natural choice when planning for the EV, it does not lead to viewpoint-invariant predictions for SVs (Hagedorn *et al.*, 2024). In addition, it complicates the modeling of interactions among SVs. For instance, this coordinate frame makes it easy to infer which vehicles are close to the EV or in front of it (small magnitude of x position). However, finding SVs that are close to each other is a second-order relation that is harder to learn. This is why other models (Janjoš *et al.*, 2022c; Jia *et al.*, 2022; Gao *et al.*, 2020; Cui *et al.*, 2019) process a scene from the viewpoint of each agent. Thus, the processing is viewpoint-invariant and makes it easier to represent interactions between all agents. On the downside, these models cannot use a shared representation for prediction and planning, and computational complexity scales linearly with the number of agents (Ngiam *et al.*, 2022). Moreover, per-agent coordinate frames are unsuitable for joint prediction, as each agent is processed individually.

An alternative way to achieve viewpoint invariance is the pairwise relative coordinate system introduced in GoRela (Cui *et al.*, 2022), which describes the relation between

agents instead of using a fixed coordinate frame. This allows the computation of the relation between static objects offline and greatly reduces computation. Pioneering Equivariant Planning (PEP) (Hagedorn *et al.*, 2024) further reduces computation by using center-relative instead of pairwise relative coordinates, i.e. shifting the coordinate frame into the mean position of all processed agents before each transformation. This makes it easier to represent and model interactions. It also eases the integration of prediction and planning, as both can use a shared scene representation. Thus, they can be combined in a single compute-efficient step.

The downside of the Cartesian frames mentioned above is that the models are vulnerable to distributional shifts and often produce inadmissible offroad trajectories (Bahari *et al.*, 2022; Hallgarten *et al.*, 2023b). Some works counter this with loss functions that penalize offroad driving (Ridel *et al.*, 2020; Messaoud *et al.*, 2020; Niedoba *et al.*, 2019; Boulton *et al.*, 2020), driving direction compliance (Greer *et al.*, 2021) or being distant from the centerline (Casas *et al.*, 2020b).

An alternative approach is to use Frenet representations that decompose positions into progress along and lateral offset from a lane (Houenou *et al.*, 2013; Yao *et al.*, 2013; Gilles *et al.*, 2022; Zhang *et al.*, 2021a). While this improves map compliance (i.e., reduces offroad), it impairs interaction modeling due to the non-euclidean coordinate frame. Moreover, this coordinate frame cannot be shared across multiple agents.

Output Representations

As suggested in Fig. 2.2, the interfaces between modules are crucial for the overall system performance. To optimize an automated driving system for the final planning task, differentiable interfaces are required (Hu *et al.*, 2023c; Karkus *et al.*, 2023). To provide interpretability, the interfaces should comprise a human-readable representation (Zeng *et al.*, 2019). When predicting trajectories in modular E2E systems, the output must fulfill these two criteria. Additional difficulty arises from the inherent uncertainty of future prediction (Hubmann *et al.*, 2018; Chen *et al.*, 2024). Since the intentions of SVs are unknown, multiple future modalities must be considered to make a safe plan (Rhinehart *et al.*, 2021). Therefore, the multimodality of P_{pred} is commonly expressed as an explicit density function or estimated with a discrete trajectory set.

Explicit density functions model the probability density function of P_{pred} (Zeng *et al.*, 2019; Sadat *et al.*, 2020; Cui *et al.*, 2021; Casas *et al.*, 2021). Continuous distributions like bi-variate Gaussian distributions (Luo *et al.*, 2020), Gaussian Mixture Models (Shi *et al.*, 2022a; Knittel *et al.*, 2023; Huang *et al.*, 2023b; Liu *et al.*, 2024), or discrete rasterized heatmaps (Gilles *et al.*, 2021a, 2022) are used in many prediction modules. Alternatively, object-agnostic representations like occupancy maps (Kim *et al.*, 2017, 2022), or flow fields (Mahjourian *et al.*, 2022; Agro *et al.*, 2023; Casas *et al.*, 2021) can be used. Object-agnostic representations handle multimodality implicitly (Zeng *et al.*, 2019) and were shown to be more robust against perturbations (Rhinehart *et al.*, 2018a). Further, density functions are interpretable for humans. However, since no discrete trajectories are

decoded, it is difficult to compare such representations with expert logs for performance evaluation.

In an alternative approach to explicit density functions, discrete trajectory sets are either created by sampling from an intermediate distribution or directly output by the model. To sample a diverse trajectory set from an intermediate distribution, generative models such as Generative Adversarial Networks (GANs) (Goodfellow *et al.*, 2020), Conditional Variational Autoencoders (CVAEs) (Sohn *et al.*, 2015), normalizing flows (Rezende and Mohamed, 2015) or denoising diffusion (Jiang *et al.*, 2023b), can be used (Lee *et al.*, 2017; Zhao *et al.*, 2019; Sriram *et al.*, 2020; Choi *et al.*, 2021; Dendorfer *et al.*, 2021). Alternatively, recent methods apply several prediction heads to the features extracted by a common backbone to decode a set of diverse trajectories (Varadarajan *et al.*, 2022; Liang *et al.*, 2020; Wang *et al.*, 2022c; Nayakanti *et al.*, 2022). Other ways to obtain diverse predictions are training loss functions (Zhao *et al.*, 2021b), entropy maximization (Deo and Trivedi, 2020; Aghasadeghi and Bretl, 2011), variance-based non-maximum suppression (Wang *et al.*, 2022a), greedy goal sampling (Phan-Minh *et al.*, 2020; Cui *et al.*, 2022), divide and conquer strategy (Narayanan *et al.*, 2021), evenly spaced goal states (Lu *et al.*, 2022), ensembling (Ye *et al.*, 2022), or using pre-defined anchor trajectories (Chai *et al.*, 2019; Song *et al.*, 2022; Fang *et al.*, 2020). Nevertheless, a guarantee of coverage can only be given by specifying high-level behaviors (Chen and Krähenbühl, 2022) or regions of the map that must be covered by at least one prediction (Liu *et al.*, 2021c).

2.3.2 Planning

Input Representations

The input of the planner depends on the order of modules in sequential IPPSs. If prediction is conditioned on potential EV actions, planning is executed first and takes the sparse or rasterized scene representation (cf. Sec. 2.3.1) as input. If prediction is executed before planning, the predictions are fed to the planner as an additional input in one of the formats described in Sec. 2.3.1.

In both cases, goal information for the EV is available in the planning module. It can have different forms like lane-level route information (Caesar *et al.*, 2021), high-level commands (Dosovitskiy *et al.*, 2017; Zhai *et al.*, 2023), and sparse goal points (Dosovitskiy *et al.*, 2017). There are four categories how goal information is incorporated into the IPPS: input features, separate submodules, routing cost, and route attention. Simply providing goal information as input feature yields no guarantees for goal compliance but is straightforward and widely adopted (Codevilla *et al.*, 2018a; Chitta *et al.*, 2021, 2022; Chen and Krähenbühl, 2022; Shao *et al.*, 2023b; Wu *et al.*, 2022; Scheel *et al.*, 2022; Chen *et al.*, 2019; Codevilla *et al.*, 2019; Hecker *et al.*, 2018; Hawke *et al.*, 2020; Ohn-Bar *et al.*, 2020). Separate submodules are used only with high-level commands, which serve as a switch between command-specific submodules (Codevilla *et al.*, 2018a; Müller *et al.*,

2018; Sauer *et al.*, 2018; Chen *et al.*, 2020; Chen and Krähenbühl, 2022). This can prevent the model from over-adapting to driving modes that dominate the training dataset, but requires a fixed number of high-level commands to be defined in advance. A third option to incorporate goal information is by optimizing a cost term, which comprises progress and route compliance (Sadat *et al.*, 2020; Cui *et al.*, 2021; Rhinehart *et al.*, 2018a). Balancing multiple targets in a cost term provides flexibility and interpretability but can lead to undesirable tradeoffs with safety. Lastly, route attention applies spatial attention to on-route parts of the map. This is achieved by Transformer cross-attention (Bronstein *et al.*, 2022) or by removing the off-route portion of the map after initial feature aggregation (Dauner *et al.*, 2023; Hagedorn *et al.*, 2024; Hallgarten *et al.*, 2023a; Chen *et al.*, 2023e; Huang *et al.*, 2023a).

Since sequential IPPSs consist of discernible modules, the interfaces between them have to comply with diverse requirements. On one hand they should be interpretable to understand failures and facilitate development. Rasterized (Bansal *et al.*, 2018; Chen *et al.*, 2020; Chen and Krähenbühl, 2022; Wang *et al.*, 2019b) and sparse (Renz *et al.*, 2022; Pini *et al.*, 2022; Hallgarten *et al.*, 2023a; Vitelli *et al.*, 2022; Scheel *et al.*, 2022) scene representations (cf. Sec. 2.3.1) are teherefore common intermediate representations, which are input to the planning module. Similar to the input representation used by prediction modules, a clear trend can be observed from rasters to sparse inputs (Bansal *et al.*, 2018; Zeng *et al.*, 2019; Huang *et al.*, 2023b,a). On the other hand an interface should convey all relevant information to downstream modules to cover the long-tailed distribution of potential driving scenarios. Irrespective of the order of modules, information loss is a problem of hand-crafted interfaces (Karkus *et al.*, 2023). For instance, if a vehicle is currently occluded, but its presence can be inferred from the reflection of its headlight in another car, bounding box representations (Bansal *et al.*, 2018), occupancy fields (Casas *et al.*, 2021), or affordances (Chen *et al.*, 2015; Sauer *et al.*, 2018) cannot propagate this information to downstream modules (Karkus *et al.*, 2023; Hu *et al.*, 2023c). Conversely, latent feature representations allow uncertainty propagation, enabling downstream modules to compensate for errors at earlier stages, such as undetected vehicles (Zeng *et al.*, 2019).

Planning Paradigms

When taking a closer look at the planning module, three general planning paradigms can be observed. We introduce each paradigm and discuss its suitability for the integration of prediction and planning. Recall that in Eqns. 2.5 and 2.6 we defined planning to be a function f mapping from observational inputs X_{EV}, $\overline{\mathbf{X}}_{\mathrm{SV}}$, I and possibly predictions $\overline{\mathbf{Y}}_{\mathrm{SV}}$ to a trajectory Y_{EV}. In the following, we focus on the planning function f. We decompose f into two parts: a proposal generator g, yielding a set of multiple potentially suitable trajectories $\hat{Y}_{\mathrm{EV}}^{(i)}, i = 1,..,N_{\mathrm{proposals}}$ and a proposal selector h that selects the final plan Y_{EV}

among the proposed set. Hence, we propose to represent the planning function f as

$$f = h(g(X_{\mathrm{EV}}, \overline{\mathbf{X}}_{\mathrm{SV}}, I) \tag{2.7}$$

or in case of prediction-conditioned planning as

$$f = h(g(X_{\mathrm{EV}}, \overline{\mathbf{X}}_{\mathrm{SV}}, I, \overline{\mathbf{Y}}_{\mathrm{SV}}) \,. \tag{2.8}$$

Based on this, we distinguish three different paradigms adopted for the planning task: Cost function optimization, regression, and hybrid planning.

Cost function optimization. As planning aims to find a trajectory that optimizes objectives such as safety, comfort, and progress, it is a natural approach to design and optimize a cost function that balances these potentially conflicting objectives. Cost function-based planning methods (Sadat *et al.*, 2019; Zeng *et al.*, 2019; Biswas *et al.*, 2024) rely entirely on the selection function h to find a suitable trajectory,

$$h = \underset{i}{\operatorname{argmin}} \; c\left(\hat{Y}_{\mathrm{EV}}^{(i)}\right), \tag{2.9}$$

where c is a cost function of the planned trajectory. Consequently, the proposal generator g may randomly sample feasible motion profiles (Zeng *et al.*, 2019) or obtain them by clustering real-world expert demonstrations (Casas *et al.*, 2021). Traditional non-learning-based methods rely on hand-crafted cost functions (Truong *et al.*, 2023; Werling *et al.*, 2010; Buehler *et al.*, 2009; Fan *et al.*, 2018; Montremerlo *et al.*, 2008; Ziegler *et al.*, 2014; Ajanovic *et al.*, 2018; Pulver *et al.*, 2021). However, designing a cost function that effectively balances these goals and generalizes to the long-tailed distribution of possible driving scenarios is challenging. Learning-based methods aim to address this by learning a cost function directly from expert demonstrations (Ziebart *et al.*, 2008; Wulfmeier *et al.*, 2015; Rhinehart *et al.*, 2018b). The learned cost function can be non-parametric (Zeng *et al.*, 2019; Wei *et al.*, 2021; Hu *et al.*, 2021) or comprise hand-designed sub-costs of which only the weights are learned (Cui *et al.*, 2021; Casas *et al.*, 2021; Sadat *et al.*, 2020, 2019). Cost function-based planning enables the intuitive integration with a prediction module by including predictions in the cost function (Zeng *et al.*, 2020; Cui *et al.*, 2021). Depending on the prediction output, this can be realized either by explicit collision checking (Vitelli *et al.*, 2022) or by adopting predicted density functions (Zeng *et al.*, 2019) as non-parametric cost. For example, the NMP (Zeng *et al.*, 2019) predicts a cost volume, which assigns a cost to each potential future EV position within the planning horizon. In contrast, P3 (Perceive, Predict, and Plan) (Sadat *et al.*, 2020) and MP3 (Map, Perceive, Predict and Plan) (Casas *et al.*, 2021) apply hand-designed sub-costs with learned weighting to heuristically evaluate safety, comfort, traffic rule compliance, and progress of candidate trajectories (Sadat *et al.*, 2019). In both cases, the cost function is then used to select the min-cost plan from the candidates.

Regression approaches rely entirely on the proposal generator g. The selection function h is the identity, and all burden is placed on the proposal generator g, which only generates a single proposal $Y_{\mathrm{EV}} = \hat{Y}_{\mathrm{EV}}^{(1)}$. Typically, regression-based planning is not used in sequential IPPSs but rather found in undirected IPPSs (Pomerleau, 1988; Bojarski *et al.*, 2016; Xu *et al.*, 2017; Sauer *et al.*, 2018; Hecker *et al.*, 2018; Müller *et al.*, 2018; Rhinehart *et al.*, 2018a; Wang *et al.*, 2019a; Chen *et al.*, 2019; Codevilla *et al.*, 2019; Hawke *et al.*, 2020; Chen *et al.*, 2020; Ohn-Bar *et al.*, 2020; Chitta *et al.*, 2022; Scheel *et al.*, 2022; Chen and Krähenbühl, 2022; Renz *et al.*, 2022). Unlike cost function-based planning, regression cannot include predictions explicitly. One possibility is to condition the plan on predictions as stated in Eqn. 2.8. Regression approaches directly output future waypoints (Hallgarten *et al.*, 2023a; Chitta *et al.*, 2022; Scheel *et al.*, 2022) or, more rarely, actions (Pomerleau, 1988; Bojarski *et al.*, 2016; Chen *et al.*, 2015; Xu *et al.*, 2017). Both representations can be converted into one another using a kinematic model (Rajamani, 2011; Janjoš *et al.*, 2021). Commonly, they are trained by supervised learning from expert data (Schaal, 1996), called behavior cloning (Bain and Sammut, 1995; Ross *et al.*, 2011; Daftry *et al.*, 2017). This is easy to implement but vulnerable to distributional shifts, which occur when the model reaches states not covered by the training data (Scheel *et al.*, 2022). Mitigation strategies include heuristically generated data augmentations or policy rollouts during training (Scheel *et al.*, 2022; Kumar *et al.*, 2022).

Hybrid planning describes methods that combine both ideas. First, a set of candidate trajectories is regressed by the proposal generator g. Then the best one is selected with a cost function in h (Cheng *et al.*, 2024). For instance, (Vitelli *et al.*, 2022; Pini *et al.*, 2022; Hu *et al.*, 2023b; Xi *et al.*, 2023a; Huang *et al.*, 2023b) perform collision checks on the candidate plans to rule out unsafe proposals. In contrast to cost function optimization, the generated trajectory proposals are not fixed but scene-dependent.

2.3.3 Integration

Human Leader

Most sequential IPPSs belong to the human leader integration principle (Zeng *et al.*, 2019; Rhinehart *et al.*, 2018a; Casas *et al.*, 2021; Sadat *et al.*, 2020; Zeng *et al.*, 2020; Cui *et al.*, 2021; Vitelli *et al.*, 2022; Hu *et al.*, 2023c; Ye *et al.*, 2023). Human leader IPPSs can only be realized in modular or modular E2E automated driving systems since the relation of prediction and planning is specifically engineered. The planned EV trajectory is conditioned on the predicted SV behavior as depicted exemplarily in Fig. 2.3. This means the influence of the EV's plan on the SVs is not modeled. Due to this unidirectional dependency, the EV shows a reactive behavior, which can lead to underconfident plans (Rhinehart *et al.*, 2021). In the lane change example in Fig. 2.3, the EV tries to find a plan that is suitable given the three predicted SV behaviors without being aware that it can influence them. Consequently, it will favor the conservative plan and fail to reach its goal lane. If the EV could reason about the influence of its own

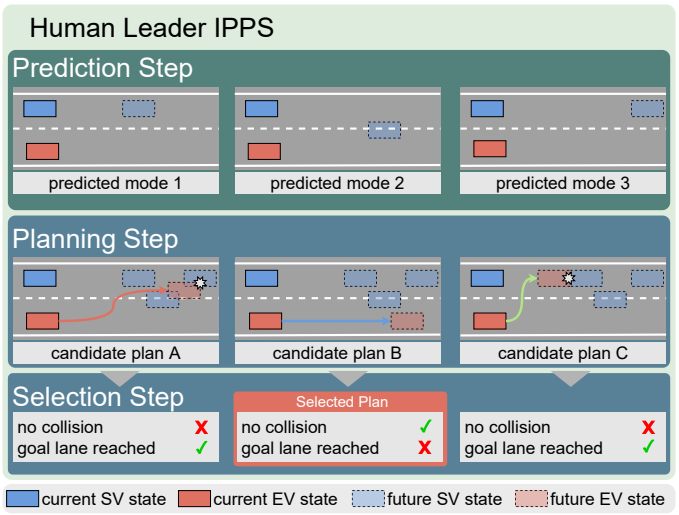


Figure 2.3: An exemplary sequential IPPS that falls in the human leader category. First, the SV's future is predicted, and then a plan is made based on the anticipated future. In this example, the planning step involves evaluating three candidate plans and selecting the one that does not collide with the predicted SV positions.

actions on the predicted SV behavior, it could anticipate that the SV might slow down or speed up further in reaction to its behavior, which would make the lane change maneuver possible. This simple example demonstrates the limitations of unidirectional, sequential IPPSs and stresses the advantage of predictions that depend on the EV's behavior.

There is a vast body of literature on models that use a human leader sequential IPPS. Deep structured self-driving Network (DSDNet) (Zeng *et al.*, 2020) uses a modular E2E system architecture. Each module builds on the outputs of the preceding module and can additionally access the latent feature of the perception backbone. Within the IPPS, a set of potential future trajectories is predicted for each detected vehicle. The planning module then samples candidate EV trajectories and applies a traditional hand-crafted cost function. The cost includes the collision probability based on the SV predictions. In contrast, the Neural Motion Planner (NMP) (Zeng *et al.*, 2019) is based on a learned cost function. Features of a shared backbone network are fed to a prediction module that forecasts future bounding boxes and a planning module that generates a spatio-temporal cost function. As in DSDNet, planning involves sampling feasible trajectories, evaluating them with the cost function, and selecting the best one. Beside P3 (Sadat *et al.*, 2020), LookOut (Cui *et al.*, 2021), and MP3 (Casas *et al.*, 2021) are further examples of this development. Specifically, P3 and MP3 predict an occupancy map while LookOut (Cui *et al.*, 2021) predicts discrete trajectories. Then a cost function that combines rule-based and learned components is used to evaluate pre-defined candidate plans. In comparison, SafetyNet (Vitelli *et al.*, 2022) follows a unique approach. A monolithic E2E planner with implicit prediction (cf. Sec. 2.4.1) is applied in parallel to a module that outputs explicit predictions. Collision

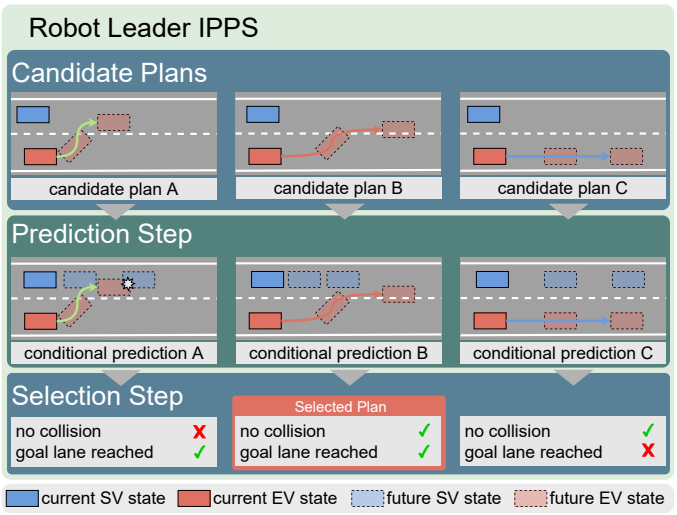


Figure 2.4: An exemplary robot leader IPPS. First, three candidate plans are generated, e.g., with a kinematic sampler. Then, a prediction conditioned on each candidate plan is inferred. In the example, the EV expects the SV to brake for it if it slowly merges to the left lane. Thus, it opts for this plan. Selecting this plan without hedging against the risk that the EV does not brake as expected results in overconfident behavior.

checks between the EV plan and SV predictions are performed to ensure the plan's safety. If the initial plan is classified as unsafe, a hand-crafted fallback layer generates a lane-aligned trajectory. Hence, the prediction module affects the selected plan, and we classify SafetyNet as a human leader IPPS. UniAD (Hu *et al.*, 2023c) implements a completely learning-based modular E2E system, which employs queries as the interface between the submodules. Thus the planning module can attend to agent-level features from previous network layers. Various works follow a similar approach: Vectorized Autonomous Driving (VAD) (Jiang *et al.*, 2023a) and Hydra-MDP (Li *et al.*, 2024b) use the output features of previous layers explicitly for collision and map compliance checks during training.

Robot Leader

In robot leader IPPSs, candidate EV plans are inferred based on the current state, and the prediction step for the whole environment is conditioned on it (Song *et al.*, 2020). For instance, (Bae *et al.*, 2022; Sheng *et al.*, 2022; Song *et al.*, 2020) predict SV trajectories conditioned on candidate plans for the EV with a neural module and then select the best one with a hand-crafted cost function. Intuitively, these methods forecast different future scenarios based on pre-defined candidate plans. Like human leader IPPSs, they can only be realized in modular or modular E2E automated driving systems due to the manually engineered influence of EV and SVs. In contrast to human leader IPPSs, the EV can anticipate the reaction of SVs to its own actions in robot leader IPPSs. Hence, the EV

can effectively seek to make the SV react to its trajectory, which can result in aggressive driving behavior (Sadigh *et al.*, 2016; Schmerling *et al.*, 2018; Tang and Salakhutdinov, 2019). Consequently, in the example in Fig. 2.4, the EV might reason that when it follows the fast-progressing plan B, the observed SV will slow down to prevent a collision.

Overall, robot leader IPPSs are no common choice in AD (cf. Fig. 2.7). In a pioneering work (Sadigh *et al.*, 2016), classic optimization is mixed with learned elements to model the effect of EV behavior on SVs. The first fully learning-based method applies a conditional variational autoencoder to obtain diverse rollouts of the future of the scene recurrently (Schmerling *et al.*, 2018). At each timestep of the rollout, the prediction is conditioned on the next step of a pre-defined candidate plan. Evaluating multimodal predictions for each candidate plan with a hand-crafted cost function yields the best plan.

The example in Fig. 2.4 reveals that robot leader systems can, other than human leader systems, anticipate the reaction of SVs to EV actions. However, this paradigm might lead to overconfident behavior. Altogether, this section has shown that no sequential IPPS can account for the bidirectional influence of EV behavior on SVs and vice versa. To resolve the issue, other integration principles are necessary.

2.4 Undirected IPPS

In contrast to sequential IPPSs, which are separated into various submodules, monolithic E2E systems employ a single neural network. Implicit systems directly map the state inputs $\overline{\mathbf{X}}_{\mathrm{SV}}$, X_{EV}, and I to the planned EV trajectory Y_{EV} (Muller *et al.*, 2005). Alternatively, joint optimization methods perform prediction and planning in a single step. Alongside Y_{EV} they output SV predictions $\overline{\mathbf{X}}_{\mathrm{SV}}$. In the following, we first examine implicit systems and then discuss joint optimization approaches.

2.4.1 Implicit

Reasoning about the future outcome of the environment is crucial when planning the behavior of the EV. Hence, it can be assumed that even models without a dedicated prediction output anticipate the behavior of their environment. Generally, the expert's actions, which the E2E planner learns to imitate in training, are based on reasoning of how the future will unfold. By learning to imitate actions, the model could exhibit behavior that is implicitly based on such reasoning as well. For instance, consider a scenario where the EV's leading vehicle brakes. An expert or a human driver will base their driving decision on the likely future scenario that the safety margin will shrink and thus brake as well. Hence, an E2E planner that imitates the expert's behavior makes a decision that is implicitly based on a predicted future. For instance, HydraMDP (Li *et al.*, 2024b) is a monolithic system that plans by regressing a score for each candidate trajectory within a predefined set. In training, an auxiliary collision cost is predicted for each candidate. Predicting this accurately is impossible without reasoning about the future behavior of

SVs. Thus, the SVs' future behavior and the interactions among them, as well as between the SVs and the EV, are modeled implicitly.

Pioneered by ALVINN (Pomerleau, 1988), there is a vast body of literature on IPPSs based on monolithic E2E planners (Bojarski *et al.*, 2016; Xu *et al.*, 2017; Codevilla *et al.*, 2018a; Hecker *et al.*, 2018; Müller *et al.*, 2018; Rhinehart *et al.*, 2018a; Wang *et al.*, 2019a; Chen *et al.*, 2019; Codevilla *et al.*, 2019; Hawke *et al.*, 2020; Chen *et al.*, 2020; Ohn-Bar *et al.*, 2020; Prakash *et al.*, 2021; Scheel *et al.*, 2022; Li *et al.*, 2018). The main disadvantage is their black-box nature, which makes model introspection and safety verification difficult. Moreover, understanding the limitations and their capabilities with respect to their interactive behavior is difficult. In fact, this can only be evaluated empirically in comprehensive benchmarks, which is what we outline as a direction for future research in Sec. 2.6.3. To improve interpretability, some works generate interpretable intermediate outputs such as online HD maps (Zhao *et al.*, 2021a), semantic segmentations (Chitta *et al.*, 2021), and object detections (Chitta *et al.*, 2022). These provide additional training supervision and introspection into the perception part of the system. We clearly separate this from systems that additionally output predictions. For instance, if the prediction output is branched off from an intermediate feature layer, the downstream part is focused exclusively on planning (Bansal *et al.*, 2018; Zeng *et al.*, 2019; Chitta *et al.*, 2021; Renz *et al.*, 2022), while the upstream part is relevant to all subtasks. Given that this makes planning a downstream task of prediction, the architecture would fall into the human leader category. In contrast, if prediction and planning are both outputs of the final layer, we assume that both tasks are addressed in a single step, which means that the architecture resembles a joint optimization system, discussed in the next section.

2.4.2 Joint Optimization

Joint optimization describes IPPSs that deterministically approximate a joint objective (Liu *et al.*, 2021a) under the assumption that an optimal outcome exists. The EV's plan is obtained by global optimization across all agents (Chen *et al.*, 2023d). This erroneously implies that the behavior model used for SVs is exactly known and that every traffic participant optimizes this same global objective. This becomes particularly challenging when considering goal-conditioning for the EV. If the joint prediction and planning process is conditioned on the EV's goal, it suggests that all SVs will behave in a manner that enables the EV to achieve its goal. At the architecture level, these systems predict and plan in a joint step. This can mean explicitly optimizing a learned or hand-crafted cost function or decoding a joint prediction for all agents, including the EV. In the latter case, the optimization can be an iterative process (Chen *et al.*, 2023f; Huang *et al.*, 2023b) or limited to one step (Pini *et al.*, 2022). An exemplary architecture is shown in Fig. 2.5, where a global cost function is used to identify the optimal plan for the EV from a set of possible future scenarios. The selected plan relies on the SV to break so the EV can merge. However, the EV selects this plan without considering that the SV might not behave

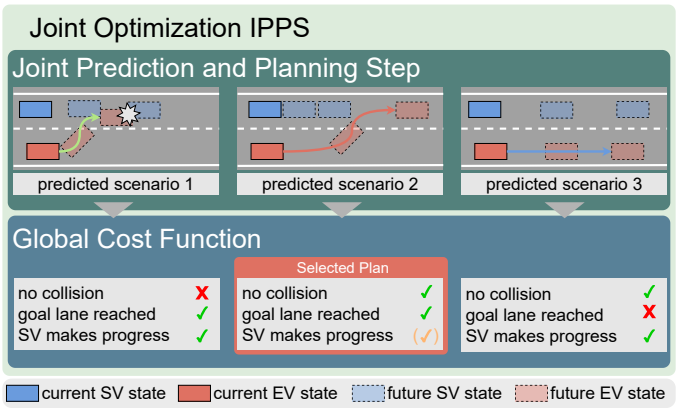


Figure 2.5: An exemplary joint optimization IPPS. A joint global cost function is optimized by selecting the best of several potential scenarios. Therefore, a joint prediction and planning step forecasts several potential scenarios that describe the EV and SV behavior. Then, each of them is evaluated using a global cost function that evaluates the outcome for all vehicles. Here, a plan for the EV is selected that corresponds to the scenario where the SV brakes to let the EV merge before it.

accordingly. In particular, joint optimization applies to monolithic planners that map all inputs to a plan for the EV Y_{EV} and the predicted future of the SVs $\overline{\mathbf{X}}_{\mathrm{SV}}$. As only the EV plan is propagated to a downstream controller and the predictions are not used further, they act as a regularization which can make the learning process more sample-efficient and can improve generalization capabilities (Bansal *et al.*, 2018; Chen and Krähenbühl, 2022; Sauer *et al.*, 2018) as it was previously demonstrated for other auxiliary tasks in prediction and planning like and occupancy forecasting (Sadat *et al.*, 2020).

A seminal work in this area is GameFormer (Huang *et al.*, 2023b). Inspired by hierarchical game-theoretic frameworks (Hang *et al.*, 2021; Wang *et al.*, 2022b; Li *et al.*, 2017; Geiger and Straehle, 2021), it models interaction between agents as a level-k game (Wright and Leyton-Brown, 2010; Costa-Gomes *et al.*, 2009). Thus, a Transformer decoder models the interaction between all agents, including the EV, by iteratively updating the individual predictions based on the predicted behavior of all agents from the previous level. Similarly, SafePathNet (Pini *et al.*, 2022) employs a Transformer module for joint prediction and planning of multiple scenarios and then ranks them based on their estimated probability and collision avoidance. Generative end-to-end autonomous driving (GenAD) (Zheng *et al.*, 2024) adapts a generative decoder to regress SV and EV trajectories jointly. The main problem with this approach is that there is no guarantee that the SVs will behave accordingly. Demonstrations show how this can lead to fatal errors (Rhinehart *et al.*, 2021). However, in applications with vehicle-to-vehicle or vehicle-to-infrastructure communication, intentions can be communicated between agents, and global optimization is a valid modeling assumption. There, a global cost function reflecting each actor's individual goals is optimized either by a central server or

by negotiation among agents. The resulting plan is assigned to each vehicle leading to more efficient and safe traffic (Klimke *et al.*, 2022, 2023).

2.5 Bidirectional IPPS

We term systems that react to anticipated behaviors of SVs but also expect SVs to react to their own actions bidirectional IPPSs, as they model the influence of prediction on planning and vice versa. Intuitively, this causes a chicken-and-egg problem, where predictions are needed to plan an appropriate behavior, and at the same time, the EV's future plan is needed to predict the SVs' reactions to it. In this section, we highlight how such systems can be realized by predicting a reactive policy instead of a fixed behavior for SVs. The resulting interactive behavior is termed co-leader.

2.5.1 Co-leader

Co-leader planning sets the highest requirements for an IPPS. It must model the influence of SVs' potential future behaviors on the EV plan as well as their stochastic reaction to potential ego trajectories (Galceran *et al.*, 2017; Fisac *et al.*, 2019; Rhinehart *et al.*, 2021). The result is an interaction-aware system that can model the anticipated behavior of its surroundings and how it can influence it. However, it does not assume this reaction to be deterministic, which is in contrast to joint optimization. Thus, the EV cannot be certain about SVs' reactions and must be prepared to react to multiple future outcomes (Cunningham *et al.*, 2015). For instance, in the opening example in Fig. 2.1, the EV can try to approach the intersection to provoke a reaction of the SV. This reaction can either indicate that the SV is going to make a turn, e.g., with a turn signal or by slowing down, or the opposite, i.e., accelerating or keeping its speed, to indicate that the SV will continue straight. The former will enable the EV to confidently decide in favor of the faster-progressing plan. Even though this paradigm is most capable in theory, only a few works exist that fulfill the challenging requirements set to the system.

IPPS Architectures

Contingencies from Observations (CfO) (Rhinehart *et al.*, 2021) originally introduced this paradigm, leveraging normalizing flows to represent multi-agent behavior as a learned coupling of independent variables. However, the intricate nature of this architecture may pose obstacles to wider adoption. Furthermore, their benchmark is simplistic, encompassing only a few scenarios with hard-coded agent behaviors. Alternatively, tree-structured methods overcome the circular dependence of prediction and planning by representing future outcomes in a tree, where each node is a state of the EV and its surroundings. Multimodality caused by different EV decisions or multiple potential

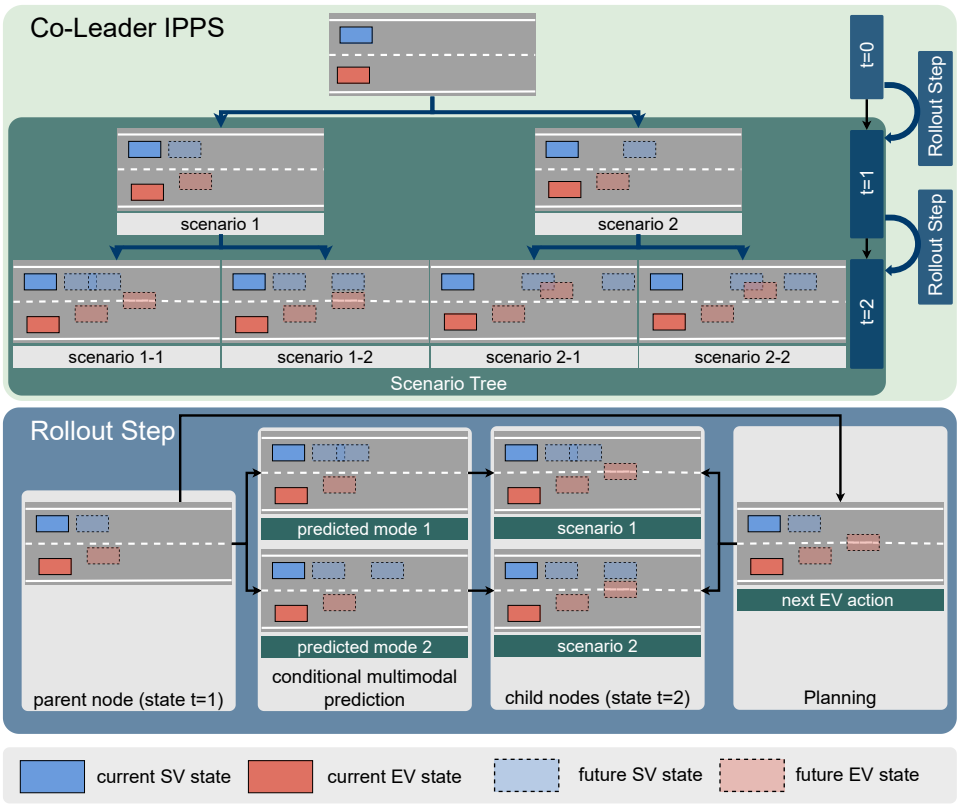


Figure 2.6: **Exemplary tree-based co-leader planner.** Starting at an initial state ($t = 0$) a tree is constructed to model potential EV plans and SV behaviors. The tree spans across several stages, each reflecting a discrete timestep within the planning horizon. In each stage, child nodes are generated by predicting the SVs and unrolling one EV action, both conditioned on the state of the parent node. This is visualized in the "rollout step" section. Note that the state in the parent node ($t = 1$) includes how far the EV has progressed along the candidate plan until this point. Thus, the prediction can only observe the candidate plan up to $t = 1$. It is not conditioned on the EV's next action, i.e., it's causal. Due to space limitations, we only visualize a prediction with two modes and only a single EV plan per rollout step. In practice, a tree might comprise more predictions and evaluate multiple potential EV behaviors. Based on the anticipated outcome of each EV behavior, i.e. the evaluation of each node w.r.t. collisions, progress and comfort, the best action can be chosen by the planner.

reactions of SVs is handled by branching (Zhang *et al.*, 2020; Ding *et al.*, 2021a). Tree-structured policy planning (TPP) (Chen *et al.*, 2023e) presents a readily comprehensible tree-structured framework that implements co-leader planning. It involves sampling kinematically feasible trajectories for the EV and then conditioning prediction on each of them. An autoregressive formulation for prediction ensures causality: it can be seen as a way to roll out the scene with SV predictions and the EV plan such that both can only access the observation history up until the respective rollout step. Autoregressive decoding can help to enforce high-quality social interactions and scene understanding (Rhinehart *et al.*, 2018b; Ivanovic and Pavone, 2019; Salzmann *et al.*, 2020; Liu *et al.*, 2021c; Yuan *et al.*, 2021; Cui *et al.*, 2022) but might lead to compounding errors since predicted information is used to make further predictions (Ross *et al.*, 2011). Finally, the best EV trajectory proposal is selected with a dynamic programming algorithm. Differentiable tree-structured policy planning (DTPP) (Huang *et al.*, 2023a) extends this with a Transformer-based conditional prediction module, which allows the forecast of all agents conditioned on multiple EV plans at once. It also replaced the dynamic program with a learnable cost function and applied node pruning to reduce computational burden. Still, the EV reaction to the prediction is taken over entirely by the cost function because the sampled ego-proposals are non-reactive with respect to the SV predictions. In general, representing the potential EV and SV behaviors in a temporal tree is easy to implement and interpret. An exemplary tree-based co-leader planning architecture similar to TPP and DTPP is shown in Fig 2.6. It can be seen that the future planning horizon has to be discretized into a limited number of steps that refer to the levels of the tree. Further, the figure shows, how the number of leaf nodes grows exponentially with the branching factor, the depth and the number of candidate plans. The branching factor has to be large enough to accurately reflect the uncertainty in the SVs' behavior as well as the behavior options of the EV and the number of candidate plans can be very high if a dense kinematic sampler is used. Moreover, discretizing the future horizon over which a scenario unfolds introduces a sensitive hyperparameter. The discretization timestep is subject to a tradeoff between runtime and accuracy w.r.t. capturing all relevant intermediate states. To counter this, TPP (Chen *et al.*, 2023e) and DTPP (Huang *et al.*, 2023a) build a tree with very limited depth of 2 stages. TPP uses a branching factor of 4, while DTPP forecasts 5 states in the first stage and branches each into 6 in the second stage. This makes architectures, which are able to model how the future unrolls without relying on discrete timesteps a promising opportunity for future research.

Scenario-Based Trajectory Planning

Bidirectional IPPSs, i.e., co-leader planning methods, are more complex than sequential or undirected IPPSs because they anticipate and consider multiple future scenarios and to make plans based on this. For instance, sequential IPPSs can be used with unimodal predictions or employ a single representation for all scenarios (e.g., occupancy grids), whereas co-leader planning methods must model the multimodal stochastic behavior of

their surroundings and plan accordingly. Similarly, monolithic E2E IPSSs with undirected integration (cf. Sec. 2.4) do not explicitly consider multiple future scenarios. Hence, the representation marginalizes all future outcomes. While this is easy to realize, it can be safety-critical. In the example of Fig. 2.1, the cost function must balance high progress and the potential safety risk if the unlikely prediction occurs for one plan with low progress for the other. Hence, it must trade off unlikely but dangerous scenarios (e.g. collision) against likely low-cost scenarios. Conversely, a co-leader architecture must make plans in the face of the uncertain reaction of SVs to its plans. In the following, we will discuss methods that incorporate multiple outcome scenarios when choosing among various EV plans. We denote the number of relevant future scenarios with N_s. They can be reflected by a multinomial distribution $P\left(\overline{\mathbf{Y}}_{\mathrm{SV}}^{(i)}\right), i = 1,..,N_s$ describing the behavior of SVs and the EV in each scenario. Based on this, we describe two groups of existing methods: Worst-case planning and contingency planning. Their difference lies in the cost function component h of the planning function $f = h(g(X_{\mathrm{EV}}, \overline{\mathbf{X}}_{\mathrm{SV}}, I)$, which selects one plan among a set of potential plans generated by g. In the architecture described in Fig 2.6, g is the tree generation step (green). The selection function h would evaluate all nodes and select the most suitable plan (not visualized).

Worst-case planning refers to IPPSs that evaluate each plan based on the worst-case outcome scenario. In the tree-based example, this would mean that if there is a collision in one node, then all nodes that share the same action would be discarded. No probabilities $P(\overline{\mathbf{Y}}_{\mathrm{SV}}^{(j)})$ are considered. The motivation behind this assumption is that the final output trajectory must be safe in each possible scenario. Consequently, the selection function h can be formulated as

$$h = \underset{i}{\operatorname{argmin}} \max_{j} c(Y_{\mathrm{EV}}^{(i)}, \overline{\mathbf{Y}}_{\mathrm{SV}}^{(j)}). \tag{2.10}$$

This paradigm strongly focuses on collision avoidance and can result in overly conservative behavior, as shown in Fig. 2.1, where only the slower progressing plan is safe under all predictions. Thus, it is preferred under worst-case assumptions.

Contingency planning aims to improve this by additionally taking into account the probabilities $P(\overline{\mathbf{Y}}_{\mathrm{SV}}^{(j)})$ of different future scenarios $\overline{\mathbf{Y}}_{\mathrm{SV}}^{(j)}$ (Hardy and Campbell, 2013; Li *et al.*, 2023b, 2024a). The resulting plan hedges against worst-case risks while enabling progress in expectation. For instance, (Cui *et al.*, 2021; Hardy and Campbell, 2013; Zhan *et al.*, 2016) find a plan that is safe for scenarios on the short horizon (independently of their probability, i.e., worst-case) and also optimize the expected cost on the long horizon (marginalization). Thus,

$$h = \underset{i}{\operatorname{argmin}} \left(\max_{j} c_s(Y_{\mathrm{EV}}^{(i)}, \overline{\mathbf{Y}}_{\mathrm{SV}}^{(j)}) + \mathbf{E}_j[c_l(Y_{\mathrm{EV}}^{(i)}, \overline{\mathbf{Y}}_{\mathrm{SV}}^{(j)})] \right), \tag{2.11}$$

where c_s and c_l consider the short-term and the long-term part of the proposal $Y_{\mathrm{EV}}^{(i)}$, respectively. In the example in Fig. 2.1, this planner allows the most progress while still

ensuring safety. It can exploit that both candidate plans have a common short-term action. Hence, it could opt for the faster-progressing plan as it is aware that it could still re-plan and fall back to the slower one. This method can also be applied to non-bidirectional IPPSs. For instance, it can be combined with a sequential human leader IPPS, which forecasts multiple scenarios. CfO (Rhinehart *et al.*, 2021) terms this *passive contingency*, as the EV is contingent upon the SVs (Cui *et al.*, 2021; Hardy and Campbell, 2013; Zhan *et al.*, 2016; Bajcsy *et al.*, 2021). In contrast, *active contingency* refers to co-leader systems that are aware that the EV can influence the SVs' behavior behavior (Galceran *et al.*, 2017; Fisac *et al.*, 2019; Bandyopadhyay *et al.*, 2013).

2.6 Discussion and Frontiers

2.6.1 Categorization Edge-cases

In the previous sections, we described and discussed different system architectures and paradigms to integrate prediction and planning in interaction-aware autonomous systems. Here, we want to emphasize the distinction between these categories and describe corner cases that transition fluently from one category to another with minor architectural tweaks. For instance, an IPPS that directly maps observational inputs to a trajectory for the EV is clearly a monolithic IPPS that falls into the implicit integration category. Extending it with an additional decoder head that outputs predicted trajectories for SVs makes it more interpretable. For instance, NMP (Zeng *et al.*, 2019) employs such an architecture. Thus, the changed architecture consists of a shared backbone for all tasks, which is connected to a prediction decoder module as well as a downstream planner module. Therefore, this change transforms the monolithic IPPS into a modular system. As the planning step comes after prediction, the system falls into the human leader category. Similarly, extending the implicit IPPS architecture with a prediction output from the final layer means that the entire network performs prediction and planning in a joint step, turning it into a joint optimization IPPS. Similarly, the exemplary architecture presented in Fig. 2.4 can be modified: It is a robot leader IPPS because it expects the SV to react deterministically to the EV's actions, and the plan is selected with only the EV's goals in mind. However, if the cost function used in the final selection step also optimizes the SV's outcome, the architecture would fall into the "joint optimization" category.

2.6.2 Benchmarking IPPS

In general, IPPSs need to be evaluated as a system, either because no submodules can be distinguished (monolithic E2E) or because of the interdependence of the submodules.

Prediction Benchmarks

A special case are widely adopted sequential human leader architectures, where the prediction component can be evaluated independently of the downstream planning task. Various prediction datasets provide expert recordings split into training, validation, and test sets for benchmarking. The most prominent ones are nuScenes (Caesar *et al.*, 2020), WOMD (Ettinger *et al.*, 2021), Argoverse (Chang *et al.*, 2019), Argoverse2 (Wilson *et al.*, 2023), inD (Bock *et al.*, 2020), highD (Krajewski *et al.*, 2018), rounD (Krajewski *et al.*, 2020), exiD (Moers *et al.*, 2022), and Interaction (Zhan *et al.*, 2019). However, optimizing the prediction performance might not lead to better planning. In fact, some works even found that optimizing prediction metrics leads to worse planning results and, thus, worse driving performance (Huang *et al.*, 2023a). One reason for this is that prediction metrics average across all target vehicles, i.e., a prediction error for a vehicle far away has the same impact on the score as an error for a vehicle right in front of the EV. Moreover, the isolated assessment of prediction modules does not consider the interplay with other submodules. For instance, a planner might navigate the EV to states that cause severe distributional shifts and impair prediction performance. Thus, it is broadly acknowledged that IPPSs must be evaluated as a system.

Open-loop Evaluation

There are two methods to evaluate Automated Driving Systems as a whole, namely open-loop evaluation and closed-loop simulation. Open-loop evaluation is similar to an evaluation in the prediction task. It compares the planner output Y_{EV} to that of an expert planner Y_{GT} (Cheng *et al.*, 2023). However, this does not give the planner control over the EV and ignores the problems arising from compounding errors and distributional shifts (Li *et al.*, 2024c). More severely, recent work demonstrated that open-loop evaluation results do not correlate with driving performance (Dauner *et al.*, 2023; Codevilla *et al.*, 2018b).

Closed-loop Simulation

Instead of merely comparing the planner output Y_{EV} to that of an expert planner Y_{GT}, closed-loop simulations let the planner control the EV. The evaluation is based on metrics for comfort, safety, and progress. Thus, closed-loop evaluation relates better to real-world driving but requires a simulator to let the model interact with its environment. Prominent simulators are Carla (Dosovitskiy *et al.*, 2017), nuPlan (Caesar *et al.*, 2021), Waymax (Gulino *et al.*, 2024), Highway-Env (Leurent, 2018), and CommonRoad (Althoff *et al.*, 2017). Carla is a high-fidelity simulator that involves sensor simulation, whereas nuPlan, Waymax, CommonRoad, and Highway-Env operate on abstract object representations (e.g., bounding boxes) without rendering camera or lidar sensors. Many benchmark datasets were proposed for the Carla simulator (Codevilla *et al.*, 2018b; Prakash *et al.*, 2021; Zhang *et al.*, 2021b; Chitta *et al.*, 2022). However, the scenarios are purely synthetic, which questions the generalizability to the real world. In contrast, nuPlan and

Waymax follow a data-driven approach, which uses recordings from real-world scenarios to initialize the simulation. Thus, users have access to corresponding large-scale real-world datasets to train their method before deploying it in simulation. CommonRoad is compatible with several smaller datasets (Bock *et al.*, 2020; Krajewski *et al.*, 2018, 2020; Moers *et al.*, 2022; Zhan *et al.*, 2019). Highway-env is based on simplistic synthetic environments.

Long-tail Scenarios

When simulating regular driving scenes, rare and critical scenarios are naturally under-represented. This problem can be tackled with data curation (Filos *et al.*, 2020; Wu *et al.*, 2023b) or additional test case generation. These can be generated manually (Klück *et al.*, 2023; Rocklage *et al.*, 2017; Birkemeyer *et al.*, 2022; Mi *et al.*, 2021; Erz *et al.*, 2022; Hallgarten *et al.*, 2024) or automatically from recorded driving logs (Gambi *et al.*, 2022; Langner *et al.*, 2018; Hauer *et al.*, 2020). Other methods generate adversarial critical scenarios by augmenting actors within the scene (Wang *et al.*, 2021; Abeysirigoonawardena *et al.*, 2019; Althoff and Lutz, 2018; Indaheng *et al.*, 2021; Klischat and Althoff, 2019; Kothari *et al.*, 2021; Wachi, 2019; Hanselmann *et al.*, 2022) or the scene itself (Zhou *et al.*, 2020; Kong *et al.*, 2020; Boloor *et al.*, 2020; Sato *et al.*, 2021; Bahari *et al.*, 2022). Similarly, (Rhinehart *et al.*, 2021; Codevilla *et al.*, 2019) investigate human-robot interaction more closely with hand-crafted scenarios – an aspect that is uncared for in average driving situations.

Traffic Agent Simulation

Another major challenge in closed-loop evaluation is to realistically simulate the SVs. In NAVSIM (Dauner *et al.*, 2024), Waymax, and the non-reactive version of nuPlan, traffic agents follow predefined trajectories obtained from real-world recordings. However, if the EV behaves differently than the expert during recording, this can result in collisions caused by the SVs, as they do not react to the EV. Hence, this simulation is usually limited to short sequences, where the EV stays close to the expert's trajectory (Zhang *et al.*, 2022a). Carla, nuPlan, and Waymax provide a reactive driver model for the SV, such as the Intelligent Driver Model (IDM) (Treiber *et al.*, 2000; Kesting *et al.*, 2010; Derbel *et al.*, 2013; Albeaik *et al.*, 2022; Sharath and Velaga, 2020). In highway-env, traffic agents can be simulated with a driving model based on IDM+MOBIL (Kesting *et al.*, 2007), which is capable of performing lane changes. Another line of work focuses on learning-based methods to simulate realistic driving behavior (Suo *et al.*, 2021, 2023; Igl *et al.*, 2022; Xu *et al.*, 2022a; Guo *et al.*, 2023; Zhong *et al.*, 2023). In Waymax (Gulino *et al.*, 2024), users can plug in their own model to control objects in the scene.

Discussion

Despite recent advancements in scenario generation and traffic simulation, there are no prominent large-scale benchmarks to evaluate the interactive behavior of IPPSs. Real-world datasets commonly comprise a limited number of lane changes or unprotected turns, and synthetic datasets with hand-crafted scenarios don't scale to the needs of comprehensive benchmarking. Moreover, leading benchmarks do not report metrics highlighting the strengths and weaknesses of different IPPS architectures concerning their interactive behavior. We emphasize this as an important avenue for future research and hope our work inspires researchers to tackle this problem.

2.6.3 Challenges and Trends

In the following section, we discuss existing trends and future challenges in the field of IPPSs. Based on our overview of deep learning-based IPPSs and their components, we identify three challenges for future research: system design, comprehensive benchmarking, and vehicle-to-X communication. After discussing them, we draw a final conclusion.

Trends

A chronological overview and categorization of influential works in the field is given in Fig. 2.7. Clustering research trends based on this overview yields three main phases of IPPSs. Until 2019 mainly monolithic systems with implicit prediction and output regression were used. Following works since 2020 parted with regression and instead incorporated prior knowledge in interpretable cost functions. Most of them integrated unconditioned predictions into the cost function and thus followed a human leader approach, which can lead to overly cautious behavior. Many works have already modularized the monolithic system and are E2E-trainable. Notably, almost all current state-of-the-art systems are end-to-end differentiable. In a third phase, beginning at the end of 2022, the integration of prediction and planning is getting more complex. In addition to sequential systems with human leader integration, undirected systems with joint IPPSs and bidirectional systems with co-leader IPPSs are implemented more frequently. Hybrid planning is also used more frequently than before. After cost functions were initially used to evaluate fixed candidate plans, newer methods often regress the candidates based on the scene and then apply the cost function.

System Design

Our survey suggests that strictly sequential architectures cannot meet the requirements set for driving systems. Nonetheless, it remains unclear which integration architecture is most effective and if the theoretical advantages of bidirectional IPPSs can be realized in practice. Especially the interactive behavior has seen little attention so far. We believe that E2E differentiable bidirectional IPPSs offer great potential for future study.

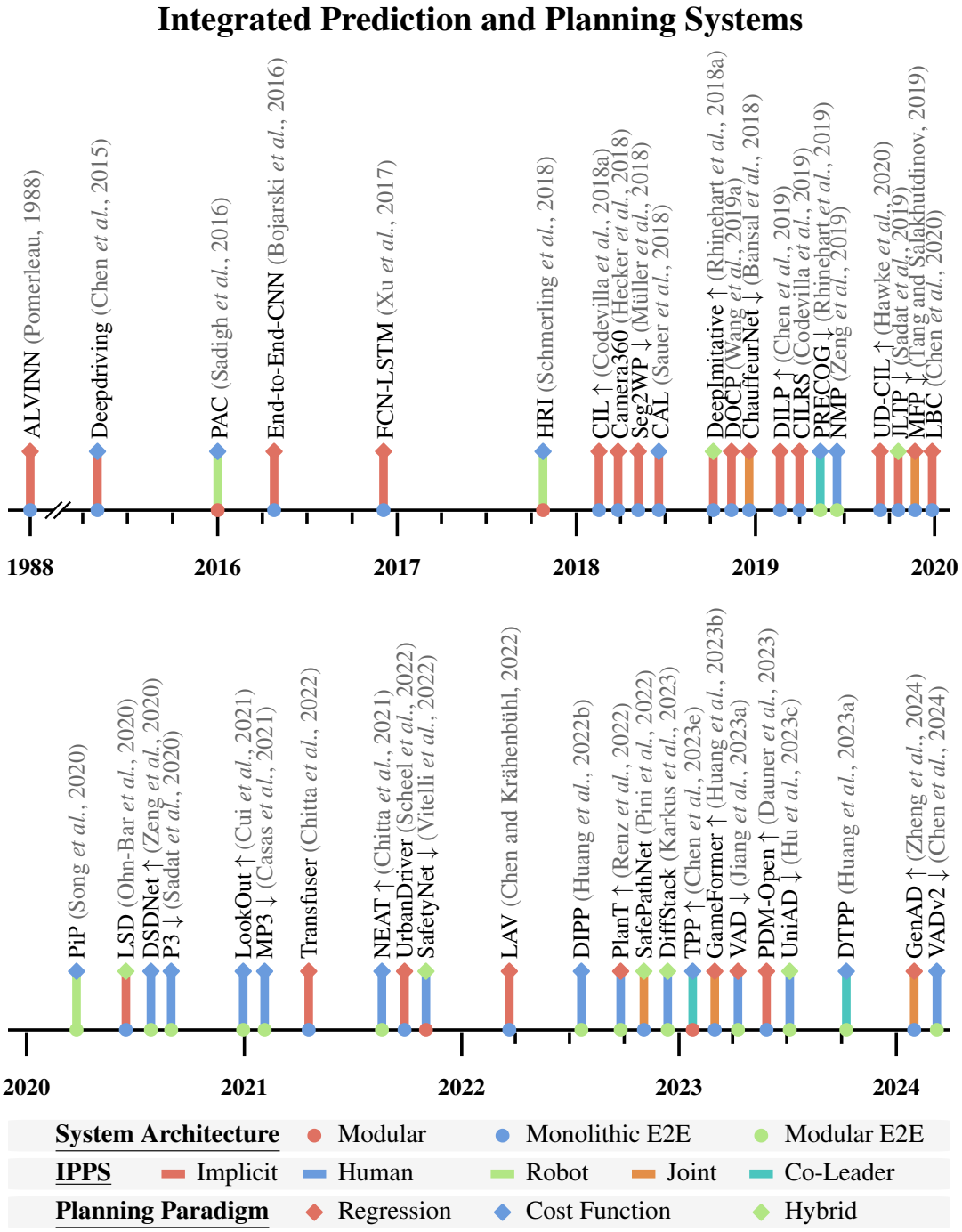


Figure 2.7: Important contributions to DL-based planning. We highlight how prediction and planning are integrated. The nodes and stem of each entry encode different aspects, which are explained below the timeline. To avoid overlaps works marked with ← and → were slightly moved into the past and future, respectively.

Benchmarking in Interactive Scenarios

We discussed different aspects of various IPPS architectures and design paradigms. However, no comprehensive empirical benchmark reproduces and analyzes their strengths and weaknesses. Such an overview would help to better understand the effects of different system architectures. This requires simulation in realistic and highly interactive scenarios with realistic driver models for surrounding vehicles and expressive interaction metrics. In this regard, data-driven simulation is a fundamental direction for future research. While powerful generative models have already been applied to scenario generation, simulating and evaluating the interactive behavior requires appropriate reactive policies to simulate traffic agents. Moreover, benchmarks must cover corner case scenarios and test the generalization to distributional shifts.

Vehicle-to-X Communication

The systems described in this chapter leverage predictions to represent the anticipated intentions of other traffic participants. With vehicle-to-vehicle or vehicle-to-X communication, these intentions don't need to be predicted but can be transmitted to surrounding agents. Thus, joint optimization methods (see Sec. 2.4.2) that can optimize several vehicles' goals can improve traffic efficiency and safety. However, such systems also need to hedge against erroneous transmissions or even malicious attacks. Moreover, they still need to be able to interact with non-communicating human-driven vehicles. Consequently, extending IPPS architectures to incorporate this additional information is a promising direction for future research.

2.6.4 Conclusion

In this chapter, we surveyed and analyzed the integration of prediction and planning methods in automated driving systems based on a comprehensive overview of the individual tasks and respective methods. We described, proposed, and analyzed categories to compare integrated prediction and planning works and highlighted implications on safety and behavior. Based on this, we discussed the shortcomings of undirected and sequential IPPSs and highlighted the need for IPPSs that model bidirectional interaction. Finally, we pointed out promising directions for future research based on the identified gaps.

Chapter 3

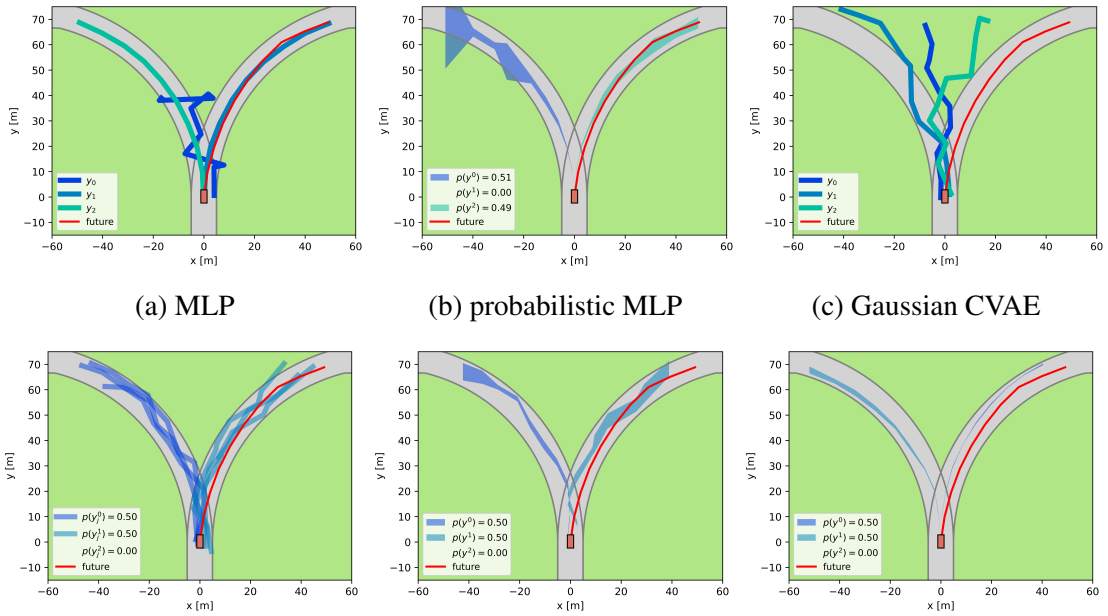
Deterministic Multimodal Trajectory Prediction

3.1 Introduction

As highlighted in the previous chapter, predicting the motion of human-driven vehicles sharing an environment with an autonomous system is a key enabler for fully automated driving. As the intention of observed vehicles and their future behavior is inherently uncertain, modeling the uncertainty in future trajectories is imperative.

Methods to address this task include discriminative encoder-decoder architectures as well as generative models. In this chapter, we propose several improvements to a special type of generative models, namely Conditional Variational Autoencoders (CVAEs). To motivate our contributions, in the following, we discuss a preliminary study to highlight their respective strengths and drawbacks of different model classes. Therefore, we evaluated several versions of an MLP-based model on a simple synthetic dataset. All samples share the same driving context: A road that branches to the left and right. We generated kinematically feasible ground truth trajectories with an action-based sampler and filtered them to prevent offroad trajectories. In total, we obtained 344 symmetric samples for each side. We direct the reader to the appendix for details on the model implementation and the dataset generation. We emphasize the limitations of our preliminary study: It does not yield a thorough evaluation of all model classes but it helps to understand the intuitions behind prevalent shortcomings and the motivation for our contributions. The results are presented in Fig. 3.1.

Deterministic Encoder-Decoder Models. Due to their simplicity and ease of use, deterministic encoder-decoder architectures are a prominent choice. While transformers or diffusion models exhibit impressive performance in public benchmarks, their runtime is prohibitive to real-world deployment. Thus, GNNs, CNNs, or MLPs are still widely used. Such models represent the uncertainty inherent to the problem by predicting a predefined number of trajectory modes using multiple decoder heads. A winner-takes-all (WTA) strategy is employed, where the loss is only applied to the predicted trajectory closest to the ground truth, and the remaining ones are ignored. As shown in Fig.3.1a, this can result in extremely sparse training feedback for single decoder heads, leading to inadmissible



(a) MLP (b) probabilistic MLP (c) Gaussian CVAE

(d) GMM-CVAE **(Ours)** (e) GMM-CVAE **(Ours)** (f) GMM-CUAE **(Ours)**

no parametric output distribution | empirical output distribution | parametric output distribution

Figure 3.1: **Comparison of model architectures in a bimodal toy example.** The top row shows established model architectures. The opacity of drawn trajectories indicates their estimated likelihood. For models with a parametric output distribution, the mean trajectory is drawn, and the width represents the standard deviation. The MLP model regresses smooth trajectories, but the WTA training pattern prevents training feedback for the mode y_0. The probabilistic MLP regresses parameters for a Gaussian Mixture Model (GMM) in output space. It downweights the extra mode to probability 0.00 but yields a bad estimate for the uncertainty of the other modes. The CVAE model with a Gaussian output space (Gaussian CVAE) cannot capture the bimodal output distribution accurately. Conversely, our GMM-CVAE samples from a GMM latent space and captures the output distribution more accurately. However, the randomly drawn samples can be spurious and include offroad trajectories. A parametric representation of the output space GMM can either be estimated empirically from the samples or using the unscented transform. The latter is done in our GMM-CUAE and gives the best results.

"dangling" trajectories, which could potentially block free space for a downstream planner. Thus, more recent architectures interpret the predicted trajectories as means of a GMM and additionally regresses covariances and mixture weights (Wang *et al.*, 2023b; Liu *et al.*, 2024; Zhou *et al.*, 2022). They are trained with a WTA-NLL loss, which selects the closest GMM component with an L2 distance and then applies an Negative Log Likelihood (NLL) loss to it. We see that the probabilistic MLP in Fig. 3.1b solves the issue of "dangling" trajectories as it can downweight the probability of the additional mode 0. Nonetheless, in our experiment the predicted covariance is inaccurate, especially for longer horizons.

Generative Models for Trajectory Prediction. In contrast, generative machine learning models such as Normalizing Flows (NFs), CVAEs, Generative Adversarial Networks (GANs) capture the uncertainty with a probability distribution in the latent space. The output distribution is obtained by randomly drawing samples from a latent prior distribution and projecting them to the output space. Thus, the training feedback of each sample is applied to the entire distribution. While GANs can be notoriously difficult to train, CVAEs generally offer easier and more stable training. At the same time, CVAEs require less careful design than the invertible flow transformations in NFs and struggle less with high-dimensional data as they do not involve computing and storing Jacobians.

Nonetheless, our preliminary study in Fig.3.1c confirms a well-known issue of CVAEs with widely adopted Gaussian latent spaces: They struggle to capture disconnected output spaces. This is because the latent prior distribution and the decoder function are both continuous. To mitigate this, we propose to leverage a more expressive GMM latent space. Fig.3.1d shows how the GMM-CVAE model captures the distribution of future behaviors more accurately.

Obtaining a Parametric Representation in Output Space. However, like the regular CVAE, the GMM-CVAE represents the output distribution by randomly sampling a latent prior distribution. The randomness of the output is seen in Fig.3.1d, resulting in some trajectories being offroad and kinematically infeasible. In a safety-critical task such as AD, deterministic model behavior is crucial for efficient validation of the system. To counter this, we propose to employ the Unscented Transform (UT) to obtain representative deterministic sigma points instead of random samples from the latent space. Putting this together with our first contribution, i.e., a GMM prior, opens the opportunity to overcome another drawback of CVAEs, namely that they don't estimate a parametric representation of the output distribution. By matching the components between the latent prior and the output space, the UT estimates for the first moments of each output component, thus yielding a parametric representation of the future trajectories in the output space. While we could also empirically estimate the output distribution from the random CVAE samples (see Fig. 3.1e), the UT yields a more accurate estimate of the means and covariances (see Fig. 3.1f) and reduces the statistical error (Julier *et al.*, 2000). We term this model Conditional Unscented Autoencoder (CUAE).

Extending Encoder-Decoder Models with Probabilistic Latent-Spaces. Inspired by the findings of our preliminary study, in this chapter, we combine the powerful architecture

of state-of-the-art encoder-decoder models with expressive CVAE latent spaces. Doing so, we demonstrate the benefits of our tweaks to the CVAE framework in the trajectory prediction task.

Our contributions are as follows:

(i) We propose to leverage a GMM prior instead of a simplistic Gaussian to improve the modeling of disconnected manifolds.

(ii) We leverage the UT guarantee deterministic model outputs.

(iii) By combining both, we can obtain a parametric GMM distribution in the output space.

(iv) We demonstrate how these tweaks to the CVAE framework can be applied to a state-of-the-art predictor.

3.2 Related Work

In the following, we discuss approaches to probabilistic trajectory prediction, i.e. modeling the conditional distribution $\mathcal{P}(\mathbf{y}|\mathbf{x})$ of future trajectories $\mathbf{y}$ given a generic context $\mathbf{x}$. A popular choice is to represent $\mathcal{P}(\mathbf{y}|\mathbf{x})$ by a **set of trajectories**. Here, a fixed number of modes with associated probabilities is regressed either individually per agent (Gao *et al.*, 2020; Cui *et al.*, 2020; Gilles *et al.*, 2021a; Zhao *et al.*, 2021b; Gilles *et al.*, 2022) or jointly for all agents in a scene (Liang *et al.*, 2020; Gilles *et al.*, 2021b; Ngiam *et al.*, 2021; Janjoš *et al.*, 2022c; Cui *et al.*, 2023). Commonly, WTA loss functions only consider the predicted mode closest to the ground truth. As non-winner modes are not penalized, the predicted set can contain unrealistic and inadmissible trajectories (e.g. off-road)..

Other classes of models attempt to directly capture $\mathcal{P}(\mathbf{y}|\mathbf{x})$ into a **parametric distribution** such as a GMM. Here, trajectories are considered means of mixture components and (co)variances are learned separately (Chai *et al.*, 2019; Khandelwal *et al.*, 2020; Phan-Minh *et al.*, 2020; Varadarajan *et al.*, 2022; Knittel *et al.*, 2023). This approach models the uncertainty of the underlying problem more accurately. Moreover, loss functions can consider the entire distribution (via the NLL), increasing sample efficiency and reducing inadmissible predictions. However, these models are theoretically limited since they do not reason about the generative process of the data, i.e. $\mathcal{P}(\mathbf{x}, \mathbf{y})$.

In contrast, **generative models** such as NFs (Rhinehart *et al.*, 2019; Schöller and Knoll, 2021; Mészáros *et al.*, 2023), GANs (Gupta *et al.*, 2018; Sadeghian *et al.*, 2019; Kosaraju *et al.*, 2019; Goodfellow *et al.*, 2020; Dendorfer *et al.*, 2020), or CVAEs (Ivanovic *et al.*, 2020; Salzmann *et al.*, 2020; Casas *et al.*, 2020a; Cui *et al.*, 2021; Yuan *et al.*, 2021; Lee *et al.*, 2022), attempt to first learn a proxy for the joint data distribution and then obtain the predictive distribution $\mathcal{P}(\mathbf{y}|\mathbf{x})$. By sampling a prior and propagating the samples into the output space, they can implicitly capture rich non-parametric distributions. Among

these models, NFs have limited expressiveness for high-dimensional data distributions found in trajectory prediction as well as strict architectural constraints, although they provide tractable likelihoods. Among GANs, prevalent issues include lack of diversity and mode collapse (Salimans *et al.*, 2016), out-of-distribution samples (Tanielian *et al.*, 2020; Khayatkhoei *et al.*, 2018), and training instability (Arjovsky and Bottou, 2017; Arjovsky *et al.*, 2017). Moreover, GANs learn a continuous transformation and are unable to model disconnected manifolds (Tanielian *et al.*, 2020; Khayatkhoei *et al.*, 2018; Dendorfer *et al.*, 2021), which is often necessary in prediction. To mitigate this, MG-GAN (Dendorfer *et al.*, 2021) uses multiple generators.

Similarly, CVAE models are commonly unable to model output distributions over several disconnected modes (Rolfe, 2016). The decoder transformation is continuous, and the latent distribution capturing multiple futures is commonly modeled using a multivariate Gaussian. The approaches in (Salzmann *et al.*, 2020; Hong *et al.*, 2019) address the issue by using a discrete latent variable mapped to a GMM output. However, this limits the expressiveness of the latent space. In this chapter, we approach this problem by leveraging more expressive latent distributions, i.e., GMMs. Another pitfall of CVAEs is that propagating the latent distribution to the output space involves drawing and decoding random samples. The randomness can result in bad coverage of the true output distribution, especially with few samples. To mitigate this, DLow (Yuan and Kitani, 2020) and AgentFormer (Yuan *et al.*, 2021) employ diversity sampling techniques in training, while LookOut (Cui *et al.*, 2021) off-loads the modeling of distinct futures to a GNN decoder. In inference, LookOut (Cui *et al.*, 2021) uses only the latent mean and abandons the rich learned latent space. In contrast, we use deterministic sampling (Janjoš *et al.*, 2023c) to obtain diverse and representative samples from the learned latent space.

3.3 Method

We consider the task of modeling $\mathcal{P}(\mathbf{y}|\mathbf{x})$, where $\mathbf{y}$ are future vehicle trajectories $\mathbf{y}$, i.e. a $T\times 2$ matrix of T future positions, and $\mathbf{x}$ is a generic context. In this chapter, we demonstrate that we can improve existing feedforward trajectory prediction models by extending them with CVAE latent spaces. Thus, we start with a brief recap of the CVAE framework in Sec. 3.3.1. Then, we present two novel tweaks to CVAE architectures. First, in Sec. 3.3.2, we propose to leverage the unscented transformation as an alternative to random sampling when propagating the predicted distribution from this latent space to the output space. Second, we propose to leverage a more structured latent space that replaces the commonly used uniform prior distribution with a mixture model. An overview of our proposed improvements on the CVAE framework can be seen in Fig. 3.2.

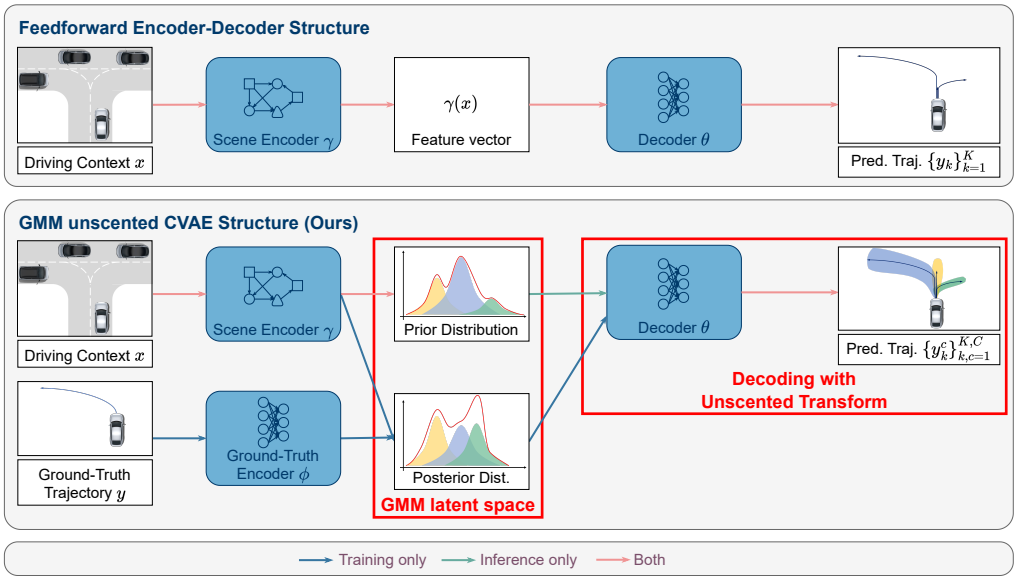


Figure 3.2: **Method Overview.** Above shows the structure of common feedforward encoder-decoder architectures that directly predict a set of K trajectories $\{y_k\}_{k=1}^K$. Below, a corresponding CVAE architecture is shown with our improvements marked in **red**. Its output is a parametric distribution over the future trajectories.

3.3.1 CVAE Background

CVAEs (Sohn *et al.*, 2015) are generative models that can capture a conditional distribution $\mathcal{P}(\mathbf{y}|\mathbf{x})$, which makes them tailored to trajectory forecasting. They model the relationship between pairs of high-dimensional inputs $\mathbf{x}$ and $\mathbf{y}$ by projecting them into a (usually) lower-dimensional latent space $\mathbf{z}$, see Fig. 3.2. An encoder parameterized by ϕ learns the latent posterior distribution $q_\phi(\mathbf{z}|\mathbf{x},\mathbf{y};\boldsymbol{\mu}_\phi,\boldsymbol{\Sigma}_\phi)$, commonly modeled as a multivariate Gaussian. Then, a θ-parameterized decoder is tasked with estimating the true output distribution $\mathcal{P}(\mathbf{y}|\mathbf{x})$. This is done by conditioning on $\mathbf{x}$ and drawing random samples $\mathbf{z}$ from q_ϕ to first produce $p_\theta(\mathbf{y}|\mathbf{x},\mathbf{z})$ and thus marginalize out $\mathbf{z}$. In inference, since the ground-truth $\mathbf{y}$ is not available, the model instead samples a surrogate, γ-parameterized latent prior $p_\gamma(\mathbf{z}|\mathbf{x};\boldsymbol{\mu}_\gamma,\boldsymbol{\Sigma}_\gamma)$. The posterior q_ϕ and prior p_γ are trained to be consistent by minimizing their Kullback-Leibler (KL) divergence. Thus, the loss function maximizes the Evidence Lower Bound (ELBO) (Kingma and Welling, 2013) and is defined as

$$\mathcal{L}_{\text{CVAE}} = \mathcal{L}_{\text{REC}} + \beta\mathcal{L}_{\text{KL}}, \tag{3.1}$$

$$\mathcal{L}_{\text{REC}} = -\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x},\mathbf{y})}\left[\log p_\theta(\mathbf{y}|\mathbf{x},\mathbf{z})\right], \tag{3.2}$$

$$\mathcal{L}_{\text{KL}} = D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x},\mathbf{y})\|p_\gamma(\mathbf{z}|\mathbf{x})). \tag{3.3}$$

The term in Eq. (3.2) promotes consistency between the decoder output and the observed ground truth, while Eq. (3.3) brings the posterior and prior distributions together by

minimizing their KL divergence. The parameter β balances these objectives. In practice, K samples $\mathbf{z}_k$ from q_ϕ are drawn and a deterministic decoder function f_θ maps each $(\mathbf{x}, \mathbf{z}_k)$ pair to an output trajectory $\mathbf{y}_k{=}f_\theta(\mathbf{x}, \mathbf{z}_k)$. Thus, no parametric form of $\mathcal{P}(\mathbf{y}|\mathbf{x})$ is estimated, and the output distribution is represented by a set of K samples. The reconstruction term in Eq. (3.2) can be approximated by the Mean Squared Error (MSE) of reconstructed samples $\mathbf{y}_k$ against the ground truth $\mathbf{y}$, yielding

$$\mathcal{L}_{\text{REC-samples}} = \tfrac{1}{K} \textstyle\sum_k^K \|\mathbf{y} - \mathbf{y}_k\| \ . \tag{3.4}$$

3.3.2 Unscented Transform of the Latent Space

Sigma Sampling A key component of the CVAE (and its Variational Autoencoders (VAE) foundation) is random sampling of the latent space. It is a feature of the reparameterization trick (Kingma and Welling, 2013), employed in order to sample the latent Gaussian prior/posterior and efficiently compute gradients w.r.t. ϕ in Eq. (3.2). However, it exhibits high variance in training and inhibits deterministic model outputs. We address this by employing a deterministic alternative based on the UT (Julier and Uhlmann, 2004) (prominent in filtering and control), which has already successfully been applied to the non-conditional VAE (Janjoš *et al.*, 2023c). Instead of sampling the prior/posterior randomly, it approximates the output distribution by selecting a set of representative points and decoding them to the output space. Computing the sigma points has proven effective in obtaining such representative points. The $2n{+}1$ sigma points $\{\boldsymbol{\chi}_i\}_{i=0}^{2n}$ of the Gaussian prior/posterior $\mathcal{N}(\boldsymbol{\mu}_\phi, \boldsymbol{\Sigma}_\phi)$, $\boldsymbol{\mu}_\phi \in \mathbb{R}^n$ are the mean $\boldsymbol{\chi}_n{=}\boldsymbol{\mu}_\phi$ and a pair on each axis $\boldsymbol{\chi}_{n\pm j}{=}\boldsymbol{\mu}_\phi \pm \sqrt{n\boldsymbol{\Sigma}_\phi}\big|_j$, $1{\leq}j{\leq}n$, where $\big|_j$ indicates the j-th column. For a commonly used diagonal $\boldsymbol{\Sigma}_\phi$, there is no computational overhead since the Cholesky decomposition $\sqrt{\boldsymbol{\Sigma}_\phi}$ can be directly obtained from predicted log variances.

Sample-based variance estimation Since the sigmas fully describe the latent distribution w.r.t. its first two moments (the mean and covariance), they usually also describe the output distribution well w.r.t. commonly used decoder nonlinearities (Julier *et al.*, 2000). Thus, we apply the UT to estimate the parameters of the distribution in output space. Therefore, we compute the sigma points of the latent distribution and then decode the corresponding trajectory for each of them. Finally, we compute the mean $\mathbf{y}_{\boldsymbol{\mu}}^{(c^*)}$ and covariance $\mathbf{y}_{\boldsymbol{\Sigma}}^{(c^*)}$ of the output distributions. Notably, if the prior and posterior are Gaussians, this step estimates a Gaussian output distribution as well. However, as shown in our preliminary study, this is insufficient to capture complex traffic scenarios with multiple (disconnected) behavior modes., which motivates our second improvement, i.e., GMM latent spaces. In practice, due to a large dimensionality n, not all sigma points are used to compute $\mathbf{y}_{\boldsymbol{\mu}}$ and covariance $\mathbf{y}_{\boldsymbol{\Sigma}}$; we select the mean sigma point and $K{<}2n{+}1$ pairs of along the covariance axis with the largest eigenvalues (Janjoš *et al.*, 2023c). We also test other heuristics to obtain a subset of sigma points in Sec. 3.4.2.

3.3.3 GMM Latent Space

The mixture prior model proposed in this section attempts to capture distinct modes of behavior using a GMM for the prior and posterior distributions in the latent space (see Fig. 3.2). Intuitively, the GMM components can correspond with modes of behavior, while the distribution of each can represent the variation within each mode. For example, one mode may correspond to a right-turn behavior whose speed or path variation is given by the variance. The prior and posterior GMMs with C components are described by $\sum_c^C w_\phi^{(c)} \mathcal{N}(\boldsymbol{\mu}_\phi^{(c)}, \boldsymbol{\Sigma}_\phi^{(c)})$ and $\sum_c^C w_\gamma^{(c)} \mathcal{N}(\boldsymbol{\mu}_\gamma^{(c)}, \boldsymbol{\Sigma}_\gamma^{(c)})$, for fixed C.

The GMM latent distribution is amendable to both methods of sampling presented so far: random samples and sigma points. In both cases samples or sigmas are drawn from each component of the latent GMM separately and then decoded into trajectory space. This can represent a more complex output distribution. To estimate a parametric representation, we estimate the mean $\mathbf{y}_{\boldsymbol{\mu}}^{(c^*)}$ and covariance $\mathbf{y}_{\boldsymbol{\Sigma}}^{(c^*)}$ of the output distributions' components empirically. In the case of sigmas, these steps represent the UT. The corresponding component weights are carried over. Consequently, we obtain a GMM representation of the output space that can represent complex distributions and lets us evaluate likelihoods.

The loss functions in Eq. (3.2) and Eq. (3.3) are adapted as follows to be compatible with a GMM representation. First, we employ an approximation for the upper bound of the KL divergence between the prior and posterior Gaussian mixtures proposed in (Do, 2003) since no analytical definition exists. It calculates the KL divergence between the discrete mixture weight distributions and the KL divergence between each corresponding posterior and prior component weighted with the posterior component weights.

$$\mathcal{L}_{\text{KL-GMM}} = \sum_c^C w_\phi \mathcal{L}_{\text{KL}}^{(c)} + D_{\text{KL}}(w_\phi \| w_\gamma) \; . \tag{3.5}$$

Second, we replace the MSE reconstruction loss with the NLL of the ground-truth trajectory under the predicted future distribution, as the MSE is agnostic to the estimated likelihood of the decoded samples. Consequently, we train the component whose centroid trajectory is closest to the ground-truth $\mathbf{y}$ (denoted by c^*) and a corresponding one-hot distribution $w_{\mathbf{y}}$ of the closest component,

$$\mathcal{L}_{\text{REC-GMM}} = -\log \mathcal{N}(\mathbf{y}; \mathbf{y}_{\boldsymbol{\mu}}^{(c^*)}, \mathbf{y}_{\boldsymbol{\Sigma}}^{(c^*)}) + D_{\text{KL}}(w_\phi \| w_{\mathbf{y}}) \; . \tag{3.6}$$

With this, we push the entire output distribution toward the ground truth instead of the individually transformed samples as in Eq. (3.4).

3.4 Experiments

In the following, we conduct an extensive experimental study to highlight the effectiveness of GMM latent spaces in combination with the unscented transform in our GMM-Conditional Unscented Autoencoder (CUAE). Therefore, we first apply our method to

Model	samples	UT	GMM prior	K	$\text{minADE}_5 \downarrow$ [m]	$\text{minFDE}_5 \downarrow$ [m]	NLL ↓ 2s	4s	6s
StarNet	✗	✗	✗	5	1.336	3.027	-	-	-
CVAE	random	✗	✗	5	2.015	4.844	-	-	-
CVAE	sigmas	✗	✗	5	1.729	3.821	-	-	-
GMM-CVAE	random	✗	✓	1	1.703	4.039	-	-	-
GMM-CVAE	random	✓	✓	65	1.419	3.358	2.100	4.163	5.658
GMM-CVAE	random	✓	✓	500	1.497	3.589	2.069	4.109	5.696
GMM-CUAE	sigmas	✓	✓	65 (all)	1.295	2.973	2.036	4.131	5.531
GMM-CUAE	sigmas	✓	✓	5	**1.273**	**2.913**	**2.009**	**4.045**	**5.420**

Table 3.1: **Ablation Study.** Left half: Breakdown of the approaches described in Sec. 3.3. Right half: Corresponding nuScenes experiment results. K is the number of samples/sigmas per component from the latent space. For the 32-dimensional latent space, a maximum of 65 sigma points can be computed, whereas an arbitrary number of random samples can be computed.

the Starnet (Janjoš *et al.*, 2022c) model and ablate the components of the GMM-CUAE separately (Sec. 3.4.1). We then showcase our method by applying it to two state-of-the-art predictors: Trajectron++ (Salzmann *et al.*, 2020) and LaFormer (Liu *et al.*, 2024). Moreover, we investigate the robustness of our method with an extensive hyperparameter study.

3.4.1 Ablation Study

In this section, we investigate the effectiveness of each component of the GMM-CUAE. We start by comparing the feedforward Starnet model to a vanilla CVAE version with unimodal prior and then successively apply the following changes: First, we replace the random sampling with using sigma points. Then, we introduce the GMM latent space and evaluate it with both random samples and sigma points.

We conduct our experiments using the nuScenes dataset (Caesar *et al.*, 2020). Given 2 seconds of historical observations of surrounding agents' bounding boxes, the model has to forecast the trajectory of a target vehicle over the next 6 seconds. We report the minimum average and final displacement error (minADE and minFDE, respectively) across five predicted modes. Note that only the models with GMM latent space are amendable to estimating a GMM output distribution based on the UT. For these, we also report the NLL of the ground-truth trajectry at 2s, 4s and 6s. Results are shown in Tab. 3.1.

We find that the vanilla CVAE (second row) performs significantly worse than the regular encoder-decoder StarNet model (first section). However, using sigma points instead of random samples improves prediction metrics significantly and, in part, closes the gap to the StarNet baseline.

Subsequently, we replace the unimodal prior with a GMM with five components, which

is shown in the last section of the table. We first draw a single random sample from each of the GMM components totaling 5 samples and decode each of them into an output trajectory. While this slightly improves Average Displacement Error (ADE) metrics, it results in worse Final Displacement Error (FDE). We then increase the number of random samples to 65 per GMM component and estimate means and covariances of the output distribution empirically. This improves prediction metrics significantly and closely approaches the StarNet baseline. Further, this lets us estimate a meaningful output distribution based on these samples and evaluate the likelihood of the ground truth trajectory. Notably, increasing the number of samples further to 500 yields only marginal improvements regarding the output distribution as seen by the NLL, but it performs worse with respect to ADE and FDE.

In contrast, the GMM-CUAE that leverages sigma points instead of random samples yields a significant improvement across NLL and displacement error metrics. Last, we reduce the number of sigma points used to estimate the output distribution to five sigmas per component. To this end, we pick the mean and the sigmas along the two axes with the largest eigenvalues. We find that this further improves the prediction performance and outperforms all baselines, including the feedforward StarNet, across all metrics. We hypothesize that picking the sigmas along the axes with the largest variance results in a diverse set of samples and prevents the mean from being overrepresented, thus preventing an overconfident estimation of the output distribution. Compared to the GMM-CVAE, which uses random samples, it gives a better estimate of the output distribution, resulting in better NLL metrics. The improvement compared to the StarNet model highlights the effectiveness of a more expressive latent prior distribution in combination with a structured sampling approach and the leverage of the unscented transform to estimate a rich output distribution.

3.4.2 Hyperparameter study

After demonstrating the effectiveness of our method, we carefully study several crucial hyperparameters of our method.

β	minFDE_5	NLL 6s
0.1	3.901	10.604
1	**2.818**	5.682
10	2.913	**5.420**
100	3.137	5.538

Table 3.2: Training objective aggregation

N_s	heur.	minFDE_5	NLL 6s
5	TEP	**2.913**	**5.420**
33	TEP	2.997	5.689
33	random	3.243	5.572
65	all	2.973	5.531

Table 3.3: Sampling methods

latent dim.	minFDE_5	NLL 6s
16	2.951	5.567
32	**2.913**	5.420
64	2.953	**5.406**
128	3.018	5.436

Table 3.4: Latent space dimension

Tab. 3.2 shows results for different values of β, the parameter that balances between the reconstruction objective (see Eq. 3.2) and the divergence of the posterior and prior distribution (see Eq. 3.3). We observe that reducing beta from 10 to 1 slightly improves

Model	$\text{minADE}_6 \downarrow$	$\text{minFDE}_6 \downarrow$
ITRA (Ścibior *et al.*, 2021)	0.17	0.49
GOHOME (Gilles *et al.*, 2022)	-	0.45
joint-StarNet (Janjoš *et al.*, 2022c)	0.13	0.38
DiPA (Knittel *et al.*, 2023)	0.11	0.34
MB-SS-ASP (Janjoš *et al.*, 2023a)	0.10	0.30
SAN (Janjoš *et al.*, 2022b)	0.10	0.29
GMM-CUAE **(Ours)**	**0.097**	**0.283**

Table 3.5: **Comparison to state-of-the-Art.** Comparison of the best model in Tab. 3.1 with models from the literature on the INTERACTION (Zhan *et al.*, 2019) validation dataset.

displacement metrics. However, this comes at the cost of significantly worse NLL. Decreasing the value further or increasing it beyond 10 yields no improvement in either metric. In addition, we evaluate different methods to obtain a subset of sigma points in Tab. 3.3. We find that using only 5 sigma points per component yields better performance than using 33 or all 65 sigma points. We also evaluate choosing 33 sigmas randomly instead of selecting them based on the eigenvalue. However, our results confirm that the top-eigenvalue-pairs (TEP) heuristic achieves the best performance. Finally, we analyze how the GMM-CUAE scales with different dimensions of the latent GMM. Starting from a 32-dimensional Gaussian per component, we test a smaller latent space with 16 dimensions. This impairs our results slightly. Increasing the size to 64 performs en-par to 32 but causes higher compute and memory demands. Increasing the size beyond 64 yields no further improvement.

3.4.3 Comparison to State-of-the-Art

Finally, we benchmark our GMM-CUAE model on the INTERACTION dataset (Zhan *et al.*, 2019), for which Starnet was originally proposed. The task is to predict the future trajectory of a target agent over the next 3 seconds, given 1 second of observation. The minimum average and final displacement errors are reported for the best of six predicted modes. The results are presented in Tab. 3.5 and highlight that our model outperforms a diverse set of strong prediction models. Qualitative samples are shown in Fig. 3.3. They highlight how trajectories reconstructed from sigma points represent more diverse behaviors. We emphasize that they also come with the benefit of deterministic and reproducible model behavior.

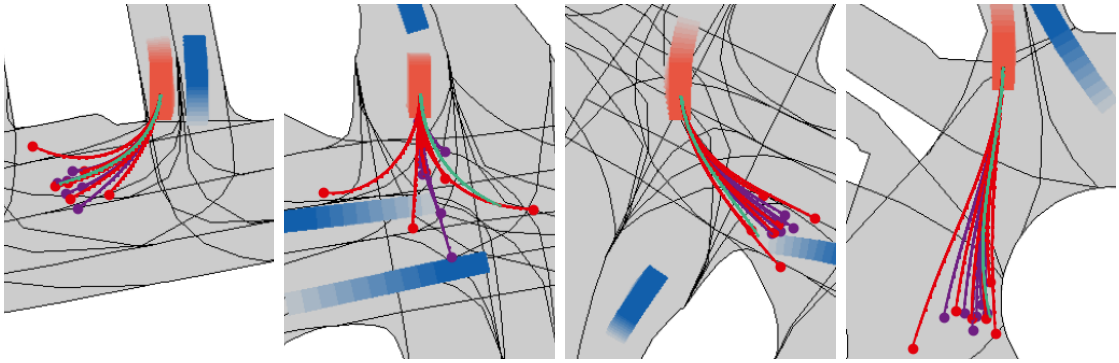


Figure 3.3: **Qualitative comparison of sampling choices.** Trajectories reconstructed from sigma points (red) are significantly more diverse than random samples (purple). Traffic participants are depicted in blue and the predicted vehicle in red, with fading for history (best viewed in color).

3.5 Discussion

3.5.1 Conclusion

In this chapter, we propose to extend feedforward encoder-decoder prediction models with a rich latent-space distribution, which is known from CVAE models. We present an extensive study that highlights how simplistic latent space distributions fall short of encoder-decoder performance. Fueled by this, we propose to leverage expressive GMMs to model a rich latent space. We show that the GMM latent space can be mapped to a trajectory output space effectively with the well-known unscented transform, a sigma point-based estimate of mean and covariance. We carefully ablate these components and show how they can be used to improve feedforward encoder-decoder predictors. Moreover, we confirmed the robustness of our proposed GMM-CUAE method in an extensive hyperparameter study.

3.5.2 Limitations

Our method is able to learn a more expressive latent space and leverage it effectively to predict accurate parametric output distributions than feedforward encoder-decoder models. Unlike these, our CVAE-based approach relies on variational inference. Despite showcasing our methods' robustness, we acknowledge that CVAEs are often difficult to train. Besides, our proposed model is based on a single-agent formulation. A general limitation of these methods is that they need to be executed for each actor individually, which can be problematic in scenes with many traffic participants. Nontheless, our CUAE framework could also be applied to joint predictors that process the entire scene at once. We highlight this as a promising avenue for future research.

Chapter 4

Map-consistent Trajectory Learning

4.1 Introduction

After demonstrating how to robustly generate accurate parametric output distributions, we now shift the focus towards another prevalent challenge in vehicle trajectory prediction, namely incorporating map information to achieve realistic motion forecasts. We start off from the observation, that state-of-the-art data-driven methods (Gao *et al.*, 2020; Ngiam *et al.*, 2019; Janjoš *et al.*, 2022c; Gilles *et al.*, 2022; Wang *et al.*, 2022a; Deo *et al.*, 2022; Liang *et al.*, 2020) achieve excellent performance on large-scale public benchmarks (Zhan *et al.*, 2019; Bock *et al.*, 2020; Chang *et al.*, 2019; Wilson *et al.*, 2023; Caesar *et al.*, 2020; Ettinger *et al.*, 2021), but have been shown to be vulnerable to distributional shifts and often fail to generalize to augmented or new scenes (Abeysirigoonawardena *et al.*, 2019; Althoff and Lutz, 2018; Klischat and Althoff, 2019; Wachi, 2019; Zhang *et al.*, 2022c). In particular, it has been shown that off-road predictions increase when the road ahead is perturbed with a curve, while the motion history and the road behind are left unchanged (Bahari *et al.*, 2022). This indicates that models strongly rely on extrapolating the motion history, which is reasonable in most scenarios (middle of corner, straight driving), but causes failures when a turn is ahead and the vehicle has not yet started to turn in. The potentially misleading correlation between observed motion history and future trajectory is a well-known problem in robot behavior prediction (Muller *et al.*, 2005; Codevilla *et al.*, 2019).

Possible solutions. One straightforward way to address this problem is to balance the training dataset so that challenging scenarios appear more frequently during training. As this would result in filtering out large parts of the learning dataset (e.g., most straight-driving scenarios) and also requires tremendous labeling efforts, it is practically infeasible. Similarly, (Bahari *et al.*, 2022) proposes to augment the training dataset with challenging samples to fine-tune the models, which merely alleviates the problem slightly. As will be shown, removing the past motion history from the inputs is also problematic, since it contains vital clues for the prediction task.

Frenet Frame Wrapper. We demonstrate how this generalization problem can be effectively addressed by leveraging a scene representation that is well-known in "classic" planning approaches: the Frenet coordinate frame. In contrast to previous works that have

used the Frenet representation in their specific model architectures (Gilles *et al.*, 2022; Zhang *et al.*, 2021a; Song *et al.*, 2022), we present a general wrapper that is applicable to state-of-the-art models without the need to make changes to the architecture. More specifically, we propose to first identify relevant lane centerlines in a traffic scene. Each of these centerlines defines a Frenet frame, for which the prediction model is queried. The model is trained to represent the prediction in the same Frenet frame, creating a strong inductive bias towards lane following. Moreover, our wrapper allows us to query a prediction model multiple times depending on the number of relevant centerlines. This promotes diversity and ensures that all relevant behaviors are captured in the set of predicted trajectories.

Contribution. In contrast to previous methods, we propose to use the Frenet representation in a wrapper compatible with state-of-the-art prediction models. As this is independent of a specific model architecture, it lets us analyze the regularising effect of the Frenet representation. We summarise our main contributions to robust motion prediction in a Frenet frame as follows:

- We propose a wrapper compatible with state-of-the-art prediction models that leverages a Frenet representation of the scene.
- We conduct experiments with two different state-of-the-art prediction models and show that our wrapper reduces off-road predictions in challenging scenarios by a large margin and highlight the tradeoff between prediction accuracy and generalization.
- We demonstrate how the additional benefit of a map-adaptive number of forecasted trajectories results in more diverse and accurate predictions.

Structure. The remainder of this chapter is structured as follows. Section 4.2 gives an overview of relevant work on trajectory prediction in the Frenet frame as well as on generalization and robustness in vehicle trajectory prediction. Subsequently, in Section 4.3 we describe our method before introducing our experimental setup in Section 4.4. Results and ablations are presented in Section 4.5 and Section 4.6 respectively. Finally, in Section 4.7 we draw conclusions and point out promising directions for future research.

4.2 Related work

Trajectory Prediction in Frenet Frame. The task of motion prediction is to forecast the future trajectory of a target vehicle target vehicle (TV) within a traffic scene, given the scene context, such as the past trajectory of the TV and surrounding road users, as well as map information. Commonly model inputs and output trajectories are represented in a Cartesian coordinate frame centered around the TV with the x-axis pointing forward and the y-axis pointing to the left (Gao *et al.*, 2020; Gilles *et al.*, 2022; Liang *et al.*, 2020; Gilles *et al.*, 2021a; Varadarajan *et al.*, 2022; Lu *et al.*, 2022; Phan-Minh *et al.*, 2020). At the same

time, the curvilinear coordinate system along a selected lane centerline, termed Frenet frame, is a natural choice to represent vehicle trajectories as it decomposes trajectories into their progress along a reference lane and the lateral displacement across it. Thus, it has been broadly adopted in planning (Pulver *et al.*, 2021; Li *et al.*, 2022a; Kant and Zucker, 1986; Werling *et al.*, 2010; Huang *et al.*, 2022b) and prediction based on heuristics (Yao *et al.*, 2013; Houenou *et al.*, 2013). In data-driven prediction, however, only a few works leverage this representation. For instance, (Zhang *et al.*, 2021a) represents the variable number of lanes in a traffic scene with a raster in the Frenet frame and subsequently models interaction between them using a GNN formulation. Finally, trajectories are decoded for each lane separately. To generate a heatmap of potential trajectory endpoints, GOHOME (Gilles *et al.*, 2022) first produces Frenet frame heatmaps for each relevant lane, projects them back to the Cartesian frame, and superposes them. Similarly, PRIME (Song *et al.*, 2022) generates feasible trajectories in a Frenet Frame (cf. (Werling *et al.*, 2010)) and then utilizes a learning-based evaluator to select a set of potential future trajectories. Unlike previous methods, our approach is not specific to a particular model architecture. Instead, we present a general wrapper that can be applied to a large variety of state-of-the-art prediction models without changing their architecture. Hence, we are able to study the tradeoff between prediction accuracy and generalisation that this representation induces.

Robustness in Trajectory prediction. Many previous works studied the robustness of data-driven prediction models, e.g. by adapting the target agent (Saadatnejad *et al.*, 2021) or surrounding agents (Abeysirigoonawardena *et al.*, 2019; Althoff and Lutz, 2018; Klischat and Althoff, 2019) in an adversarial manner, with the goal of tricking the prediction model into forecasting an unrealistic trajectory. However, it is non-trivial to define when a prediction model has been fooled in these augmented scenarios. For instance, if the TV swerves in its lane, predicting that it will drive into the opposite lane could well be reasonable. Other works focus on generating new road topologies instead of augmenting surrounding agents. For instance, (Saadatnejad *et al.*, 2021; Mi *et al.*, 2021) utilize generative models to create novel maps. However, they are not necessarily realistic, mainly due to potential artifacts. In contrast, (Bahari *et al.*, 2022) proposes to generate maps by perturbing existing scenes from a public benchmark (Chang *et al.*, 2019). In particular, the road topology is augmented by adding turns in front of the TV, while the road behind the TV and its motion history are left unchanged. The resulting maps are shown to resemble roads and intersections in the real world hence it is confirmed that these perturbations are indeed realistic. At the same time, the generated scenarios are challenging, as the model has to predict the TV to turn in, i.e. significantly deviate from the extrapolation of the past motion. It is shown that 60% of the benchmark scenes can be modified so that prediction methods fail. In this chapter, we use this benchmarking methodology to evaluate our approach and show that integrating state-of-the-art prediction models in a Frenet wrapper reduces off-road predictions by over 90%.

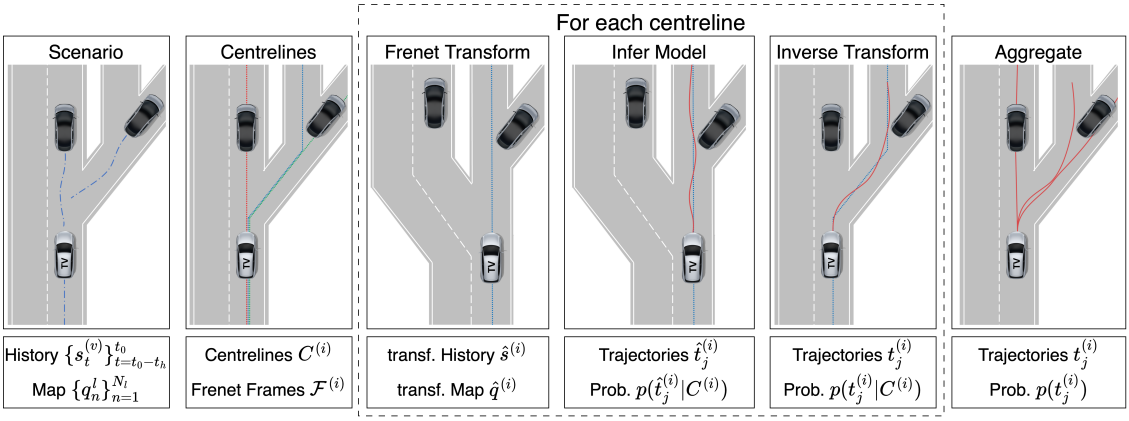


Figure 4.1: **Method Overview.** After identifying relevant lane centerline sequences for the target vehicle (TV), the scene is transformed to the corresponding Frenet frame of each centerline. Next, the prediction model is inferred, and the output is projected back to the cartesian frame. Finally, predictions from all centerlines are aggregated. For simplicity, the figure shows $K = 1$ predictions per centerline.

4.3 Method Description

4.3.1 Notation and Definitions

Prediction models forecast a TV's future behavior conditioned on the past trajectories of the TV and surrounding agents, as well as an HD Map describing the road topology. We use the notation introduced in section 2.2 and represent past trajectories for agent v as a sequence of states $\mathbf{s} = \{s_t^{(v)}\}_{t=t_0-t_h}^{t_0}$ with t_h and t_0 being the past observation horizon and current timestep respectively. Besides describing an agent's position, the state vectors $s_t^{(v)} \in \mathbb{R}^m$ may also include further attributes, such as velocity, yaw rate, or acceleration. The map is represented as a set of L lanes given by a sequence of poses on their centerlines $\mathbf{q} = \{q_n^l\}_{n=1}^{N_l}, l = 1,...,L$ where l is the index of the lane and n is the index of the pose on the centerline. The total number of poses describing the centerline l is denoted by N_l. Furthermore, lane connections are described by an adjacency matrix. A prediction model uses these inputs to forecast the future motion of a selected TV. Since it has no access to the TV's intention, the future behavior is inherently uncertain and multimodal. To address this, commonly a set of K trajectories $t_k, k = 1,..,K$ is predicted, and a probability $p(t_k)$ is assigned to each.

4.3.2 Predicting in Frenet Frames

We propose to integrate state-of-the-art prediction models in a Frenet frame wrapper. An overview of our method is shown in Fig. 4.1. It consists of the following steps: First, assign the TV to a lane centerline and identify all relevant N sequences of lane centerlines.

Second, iterate over the centerlines and transform the full scene representation into the Frenet frame of that centerline. Then query the prediction model to obtain K trajectories for each of the N centerlines, totalling $K \cdot N$ trajectories $t_k^{(n)}$ each assigned an (unnormed) probability $p_k^{(n)}$. To form the final set of output trajectories, suppress near-duplicate trajectories and select a set of $\hat{K}$ trajectories based on their probabilities.

Identification of Relevant Centerlines. We represent the model inputs and outputs in the Frenet frame using lane centerline sequences as reference lines. Given the uncertainty regarding which lane sequence the TV will traverse, we evaluate the prediction model for all of them. All sequences start at the current lane of the TV and follow different successor lanes through intersections and forks. The current lane is determined based on the distance to the lane center and orientation match. The reference line for the Frenet transformation is obtained by extending this lane's centerline with the respective successor lanes until a fixed length is reached. For our experiments, we use the closest lane with a maximum heading deviation of $\frac{\pi}{4}$ as the current lane and extend it to a length of 110 meters, which was selected empirically. At intersections and forks, we duplicate the centerline and follow each successor individually. This may yield identical centerline sequences up to a very distant point, resulting in nearly identical inputs for the prediction model. However, we decide to deal with these near-duplicates at the trajectory level, rather than at the centerline level. We denote the sequences of centerlines $C^{(i)} \in \mathbb{R}^{m \times 2}, i = 1,..N$ and the respective Frenet frame by $\mathcal{F}^{(i)}$ where m is the number of 2D, i.e., (x,y), poses along $C^{(i)}$.

Frenet Frame Transformation. We transform $\mathbf{s}$ and $\mathbf{q}$ into each Frenet frame $\mathcal{F}^{(i)}$. We chose the origin of $\mathcal{F}^{(i)}$ such that the TV's current position is at $s = 0$. Coordinate frame-independent attributes (e.g., speed) remain unchanged. Additionally, we replace the yaw angle in $\mathbf{q}$ with the curvature.

Inferring the Prediction Model. We infer the prediction model using the inputs $\hat{\mathbf{s}}^{(\mathbf{i})}$ and $\hat{\mathbf{q}}^{(\mathbf{i})}$ for each Frenet frame $\mathcal{F}^{(i)}$. The model's architecture is left unchanged. However, unlike for the Cartesian model, we interpret the output to be represented in the same frame $\mathcal{F}^{(i)}$. I.e. predicted trajectories $t_j^{(i)}$ are represented in $\mathcal{F}^{(i)}$ and probabilities $p(t_j^{(i)}|C^{(i)})$ are conditioned on the centerline $C^{(i)}$. At inference time, the trajectories are projected back to the Cartesian frame. As we query the model a varying number of times, depending on the number of identified centerlines N, we obtain a variable number of $K \cdot N$ output trajectories.

Output Aggregation. We believe that a variable number of modes is not bad per se, as the number of potential future behaviors strongly depends on the traffic scene. However, to comply with public benchmarks, we reduce the number of output trajectories to a fixed number $\hat{K}$. We estimate the prior $p(C^{(i)})$ with a simplistic MLP model that predicts a score for each centerline conditioned on the target vehicle's past motion history and a set of points along the centerline, both represented in the original Cartesian frame. A subsequent softmax layer over all centerlines yields the probability of each lane $p(C^{(i)})$, which is used to obtain the probabilities of the trajectories $p(t_i)$ by marginalizing the

predicted conditioned probabilities $p(t_i|C^{(i)})$. Similar to (Gilles *et al.*, 2021a), we then greedily pick a set of N trajectories, starting from the one with the highest probability and suppressing all trajectories whose endpoint is located within a radius of 1.0 meters. We also experiment with alternative methods such as assuming a uniform prior or using K-means clustering as in (Deo *et al.*, 2022).

Learning. We want to teach the model a strong inductive bias towards following a given lane. Since only a single ground-truth trajectory is available during training, we determine the corresponding centerline C^* based on minimal average displacement. Subsequently, we infer the model only with respect to the respective Frenet frame $\mathcal{F}^*$. We use the inputs $\hat{\mathbf{s}}^*$ and $\hat{\mathbf{q}}^*$ represented in $\mathcal{F}^*$ to obtain K trajectories t_j^* and probabilities $p(t_j^*|C^*)$. After transforming the ground truth to $\mathcal{F}^*$, we apply the same loss function as for the model trained on cartesian data.

4.3.3 Efficient Transformation Algorithm

In the following, we describe our algorithm to transform a point $P = (x,y)$ given in the same Cartesian frame into a Frenet frame defined by the centerline c, i.e.

$$(s,d) = \mathcal{F}^{(c)}(x,y)\,, \tag{4.1}$$

where (s,d) are the longitudinal and lateral coordinates in the Frenet frame, As we will discuss afterwards (cf. Sec. 4.3.3), we can vectorize this algorithm to transform a set of M points at once efficiently.

Algorithm. Our algorithm first generates a lookup table that stores the longitudinal progress p_i of each point i in c. Then, the point C_l in c, which is closest to P is calculated. The longitudinal coordinate s is then the progress of C_l, i.e., p_l. The lateral offset can be calculated as $|d| = \|P - C_l\|_2$. This makes use of the property that if C_l is the closest point to P, then the vector $P - C_l$ is perpendicular to the reference lane c in C_l. We define the sign of d to be positive if P is located on the left of c, i.e.,

$$sign(d) = sign((C_l - C_{l-1}) \times (P - C_l))\,. \tag{4.2}$$

Hence, the transformed coordinates are

$$(s,d) = \mathcal{F}^{(c)}(x,y) = (p_j,\, sign(d) \cdot \|P - C_l\|_2)\,. \tag{4.3}$$

Lookup table for longitudinal progress along path. In the first step, we calculate a lookup table that stores the progress p_i of each point i. We start by calculating the progress increments between the points on the reference path

$$\Delta p_i = \|(x_i^c - x_{i-1}^c, y_i^c - y_{i-1}^c)\|_2 \qquad \forall\, i = 2..M\,. \tag{4.4}$$

The progress of each point along the reference lane is obtained by integration of Δp_i:

$$\hat{p}_i = \sum_{j=2}^{i} \Delta p_j \qquad \forall i = 2..M \tag{4.5}$$

The progress of the first point p_1 is zero by definition. Both operations can be computed efficiently as they are parallelisable vectorised operations.

In order to clearly define a Frenet frame based on the reference centerline c, we need to define an origin on the centerline c. We chose to select the point which is closest to the target vehicle's current position as the origin. As the local Cartesian coordinate frame is centred around the vehicle, its current position is $(0,0)$. Hence, we can infer the index o of the closest point as

$$o = \underset{i}{\operatorname{argmin}} \, \|c\|_2 \tag{4.6}$$

where $\|.\|_2$ denotes an elementwise Euclidean distance. This is as well a vectorised operation that can be computed efficiently. To shift the origin to o, we calculate the progress

$$p_i = \hat{p}_i - p_o \,. \tag{4.7}$$

Matching the closest point. Our goal is to find the index l of the closest point in c to P. Therefore, we first calculate the distance between all points in c and P,

$$|d_i| = \|x_i^c - x, x_i^c - y\|_2 \,. \tag{4.8}$$

The closest point can be obtained by $l = \operatorname{argmin}_i |d_i|$.

Output. . The final transformation of P is hence given as

$$\mathcal{F}^{(c)}(x,y) = (p_l, sign(d_l) \cdot |d_l|) \,. \tag{4.9}$$

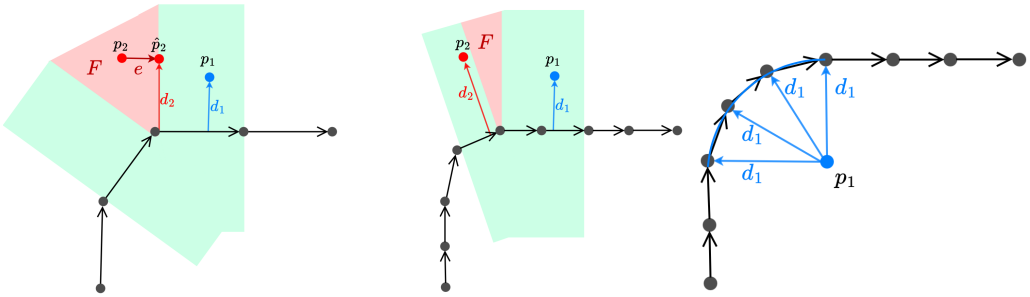
Vectorisation. The algorithm can be vectorised to efficiently transform a set of M points $P_j = (x_j, y_j)$, $j = 1..M$ in parallel, i.e.,

$$\{s_j, d_j\}_{j=1}^{M} = \mathcal{F}^{(c)}(\{x_j, y_j\}_{j=1}^{M}) \,. \tag{4.10}$$

As the centerline is identical for all points, Eqs. 4.4 to 4.7 remain unchanged. In order to calculate the closest point $C_{l(j)}$ in c for every point P_j in parallel, we first calculate the pairwise distance between all points in c and P_j

$$D_{ij} = \|x_i^c - x_j, x_i^c - y_j\|_2 \,. \tag{4.11}$$

This is the most computationally intensive step of our transformation. Still, there are relatively efficient implementations for this (Virtanen *et al.*, 2020). Then, $l(j) = \operatorname{argmin}_i D_{ij}$ denotes the index of the point in c that is closest to P_j. Finally, we get the transformed



(a) Low resolution centerline (b) High resolution centerline (c) No unique closest point

Figure 4.2: **Transformation Edge Cases**

coordinates

$$(s_j, d_j) = \mathcal{F}^{(c)}(x_j, y_j) = \left(P_{l(j)}, sign(d_j) \cdot |D_{l(j),j}|\right), \tag{4.12}$$

where

$$D_{l(j),j} = \|P_j - C_{l(j)}\|_2 \tag{4.13}$$

is the distance of P_j to the reference line.

Computational Burden in Multi-Agent Settings. In a multi-agent setting, the prediction model has to be inferred for each actor in the scene. This makes additional operations, such as the coordinate transform, expensive. However, in our case, the number of operations does not increase linearly with the number of agents. For instance, if several agents are located on the same centerline, the transformed scene representation can be reused. Only the origin has to be shifted, which is a computationally cheap operation (cf. Eq. 4.7). Thus, the number of operations depends on the number of lanes that are used as reference paths for the transformation. Hence, computational complexity does not scale linearly with the number of agents and is instead bounded by the number of relevant centerlines in the scene.

4.3.4 Handling Corner Cases

Curvature Jumps. A jump in the curvature of the reference path cause an error in the transformation. The problem is depicted in Fig. 4.2a. It demonstrates how point p_2 cannot be accurately transformed, as the distance d_2 to the closest point is not orthogonal to the curve. Green shows all points that can be transformed without error. In Fig. 4.2b., the same centerline is sampled at a higher resolution. It can be seen how the area of points where the assumption breaks becomes smaller. We make use of this and increase the resolution of the reference centerline c before applying the transformation algorithm. Thus, we fit a spline to the sequence of points and supersample it. However, the increased number of points N in the centerline increases the computational burden, especially when calculating the distances as described in Eq. 4.11. We found that supersampling the resolution by a

factor of 100 effectively balances computational overhead and transformation accuracy. In our experiments, this results in spacing between adjacent points of 0.05m.

No unique closest point. In cases where a point p_j cannot be assigned to a unique closest point $c_{l(j)}$, the transformation is ambiguous. This is demonstrated in Fig. 4.2c, where the distance of p_1 is at the center of a circular curve. Hence, its distance to multiple points on the reference lane is d_1. To resolve the ambiguity, we pick the point which is closest to the target vehicle, i.e., the one with the smallest absolute progress $|p|$. Note that this choice does not prevent accurate reconstruction.

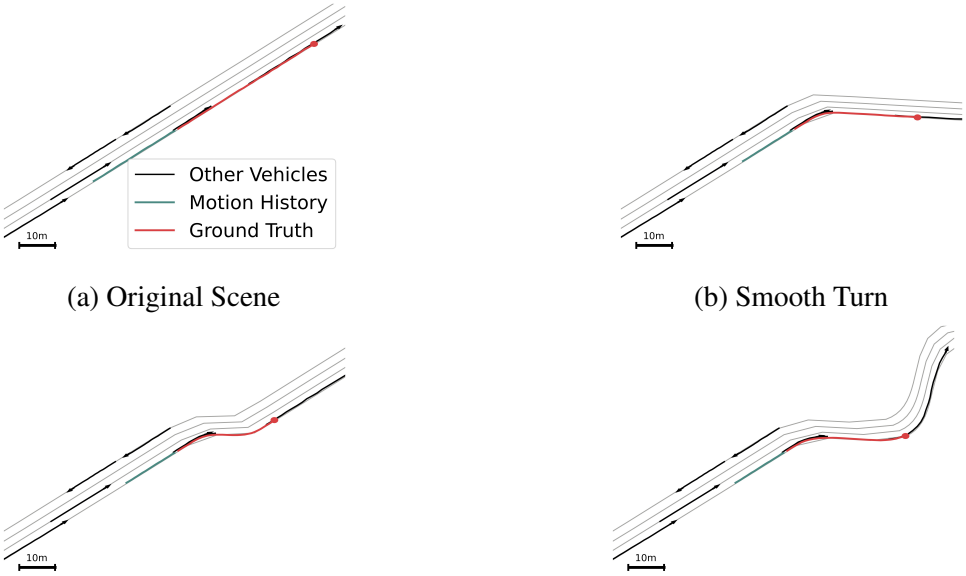
4.4 Evaluation

4.4.1 Benchmark

We evaluate our method on the Argoverse (Chang *et al.*, 2019) dataset. Collected in Pittsburgh and Miami, this large-scale dataset comprises 205,942 scenarios for training and 39,472 for testing. Each sample consists of 2 seconds of history with the goal of predicting the next 3 seconds of the TV's future motion. We leverage the scene-attack benchmark (Bahari *et al.*, 2022) to generate new challenging test scenarios and conduct experiments on the generalization ability of our method. This benchmark generates new scenes conditioned on existing scenes by shifting "scene points" according to different transforms. Scene points include the observed past trajectories of all agents, the ground truth future of the TV, and points describing the lane centerlines. The transform functions only modify scene points after a predefined distance b ahead of the TV, leaving the road behind and immediately ahead of it unchanged. To make sure the generated scenario is still feasible, velocities may be reduced such that the curve ahead can be passed safely at the current speed, i.e. without exceeding a lateral acceleration of $0.7g$. Fig. 4.3 shows the three attack functions we use to perturb an original scene: smooth turn, double turn, and ripple road. Each attack function is applied in both directions (left and right), and we report the worst result.

4.4.2 Metrics

We use the following metrics to evaluate our results. Offroad Probability (ORP) is the cumulative probability of all predicted trajectories with at least one waypoint off-road. To quantify the diversity of the predicted set of trajectories, we use Mean Inter-Endpoint Distance (MIED), the average distance of all predicted endpoints to the mean endpoint. We further report the commonly used metrics for trajectory prediction, namely minimum Average / Final Displacement Error (minADE / minFDE). Additionally, we use single-trajectory Miss Rate (MR1) to evaluate the ratio of scenarios in which the endpoint of the trajectory with the highest probability is more than 2.0 meters away from the ground truth. Since we do not have a ground truth future trajectory for perturbed scenes, we



(a) Original Scene

(b) Smooth Turn

(c) Double Turn

(d) Ripple Road

Figure 4.3: **Scene-Attack Perturbations.** Three different transformations are applied to the original scene. All of them create a turn at a fixed distance ahead of the target vehicle. Observe how the motion history and the ground truth are slowed down to account for the large curvature.

generate pseudo-labels by transforming the future trajectory in the same way as the scene points. Like the motion history, the future motion may be scaled to a lower speed to ensure physical feasibility.

4.4.3 Models

We test our wrapper approach with two state-of-the-art prediction models: LaneGCN (Liang *et al.*, 2020) and Multipath++ (Varadarajan *et al.*, 2022). Code for LaneGCN (Liang *et al.*, 2020) is publicly available and was among the best contestants in the Argoverse Forecasting Challenge 2020. Besides, we use the faithful reimplementation of Multipath++ (Varadarajan *et al.*, 2022) given in (Konev, 2022). We compare the prediction performance of the original models with those integrated into our Frenet wrapper and denote the latter with the suffix *SD* (as Frenet coordinates are commonly denoted as (s,d)). LaneGCN (Liang *et al.*, 2020) is trained for 36 epochs with a batch-size of 32. The learning rate is decayed from 10^{-3} to 10^{-4} after 32 epochs. Multipath++ (Varadarajan *et al.*, 2022) is trained for 100 epochs with a batch-size of 128, an initial learning rate of 10^{-4} that is decayed by 0.5 every 20 epochs.

4.4.4 Baselines

Off-road predictions indicate that the predictor does not consider the map information sufficiently. In particular, we notice that the predicted trajectories of the Cartesian models hardly change with the perturbations (see Fig. 4.4). Hence, we speculate that the models tend to extrapolate the motion history. We compare our method to two simple baselines that address this problem. First, we add a training regularisation that prevents the model from only extrapolating the history by applying dropouts to the TV's motion history in 50% of the training samples. We denote this with the suffix *DO*. Second, we remove the motion history entirely from the model input at training and inference time (suffix *NH*). As augmenting the training dataset with perturbed scenarios has been shown to only alleviate the off-road prediction problem slightly (Bahari *et al.*, 2022), we train neither the Cartesian nor the Frenet models with data augmentation. To emphasize that our Frenet frame wrapper surpasses a simple lane following baseline, we additionally evaluate a simple heuristic model: predicting trajectories with fixed accelerations $a \in \{-4,2,0,2,4,a_t\}\frac{m}{s^2}$, where a_t is the current acceleration. This baseline is also evaluated in the Cartesian frame (*CA*) as well as in the Frenet frame (*CA-SD*).

4.5 Results

In the following, we present our results on the original scenes from the Argoverse dataset and the scene-attack benchmark. Based on these results, we want to answer the following questions:

Model	$\hat{K}$	**minADE** ↓ [m]	**minFDE** ↓ [m]	**ORP** ↓ [%]	**MR1** ↓ [%]	**MIED** ↑ [m]
Multipath++	6	0.772	1.399	1.7	56.7	2.908
MultiPath++SD **(Ours)**	6	0.837	1.470	1.2	59.8	3.216
MultiPath++SD **(Ours)**	$N \cdot 6$	0.767	1.274	1.4	59.8	3.473
LaneGCN	6	**0.645**	**1.076**	**0.7**	**48.7**	3.800
LaneGCN-SD **(Ours)**	6	0.737	1.233	1.1	54.1	4.261
LaneGCN-SD **(Ours)**	$N \cdot 6$	0.704	1.138	1.3	54.1	**4.678**
MultiPath++DO	6	0.784	1.425	1.6	56.8	2.793
MultiPath++NH	6	1.137	2.028	1.8	75.5	4.813
LaneGCN-DO	6	**0.659**	**1.114**	0.7	**52.0**	3.889
LaneGCN-NH	6	0.983	1.665	1.0	59.7	5.729
CA-SD	$N \cdot 6$	2.410	3.745	**0.1**	–	**12.180**
CA	6	2.659	4.669	14.5	–	12.050

Table 4.1: **Results on original scenes.** SD denotes Frenet models, DO motion history dropouts and NH models without access to history.

- Does the Frenet wrapper result in predictions that are competitive with Cartesian models?
- Do models trained with our Frenet wrapper generalize beyond simple lane following?
- Can we use the Frenet representation to increase the robustness of state-of-the-art predictors?

4.5.1 Comparison on Original Scenes

We report our results on the unperturbed Argoverse benchmark in Tab. 4.1. We find that the Frenet models (suffix *SD*) with $\hat{K} = 6$ have higher displacement errors than the corresponding Cartesian models. However, these shortcomings can be partially compensated for when using the full set of $\hat{K} = N \cdot 6$ trajectories. In the case of the Multipath++ (Varadarajan *et al.*, 2022) model, the Cartesian baseline is even outperformed. At the same time, the off-road probability is of equally small magnitude and the predictions are more diverse. We observe that the increase in ORP of 0.4 percent of the LaneGCN-SD model results from scenarios where it fails to accurately follow a different lane than the reference lane. Note that an ORP of 1 percent indicates that the off-road trajectories amount to a probability of only 0.01 on average.

Our qualitative results (see Fig. 4.4) also show that the *SD* models do not simply follow the lane centerlines. On the contrary, we observe how they adapt to corner cases where a branching centerline is not detected because the TV has just moved past the branching

Model	$\hat{K}$	Single Turn			Double Turn			Ripple Road		
		minADE	ORP	MR1	minADE	ORP	MR1	minADE	ORP	MR1
MultiPath++	6	3.139	65.5	96.5	2.574	67.1	96.3	3.549	72.9	97.9
MultiPath++SD **(Ours)**	6	1.388	3.6	88.0	1.132	3.7	78.5	1.331	3.5	85.2
MultiPath++SD **(Ours)**	$N \cdot 6$	1.265	3.7	88.2	1.018	3.8	78.8	1.265	3.7	85.2
LaneGCN	6	2.063	38.2	93.9	2.372	63.9	94.4	3.094	61.2	96.2
LaneGCN-SD **(Ours)**	6	1.209	**3.1**	**85.6**	0.987	**3.5**	**70.3**	1.144	**2.9**	**81.7**
LaneGCN-SD **(Ours)**	$N \cdot 6$	**1.146**	3.3	85.8	**0.940**	3.7	70.7	**1.075**	3.0	**81.7**
MultiPath++DO	6	3.205	97.8	**56.8**	2.594	69.7	97.7	3.457	74.5	98.9
MultiPath++NH	6	5.653	98.7	75.5	3.459	73.8	97.4	4.264	78.1	98.7
LaneGCN-DO	6	**2.209**	45.7	95.2	2.312	59.6	**95.1**	2.937	57.2	**96.6**
LaneGCN-NH	6	2.652	54.7	97.1	3.150	68.6	97.0	3.705	67.8	98.1
CA-SD	$N \cdot 6$	**2.209**	**0.5**	–	**2.175**	**1.1**	–	**2.251**	**0.0**	–
CA	6	4.748	58.2	–	4.071	57.6	–	5.208	61.9	–

Table 4.2: **Results on perturbed scenes.** SD denotes Frenet models, DO motion history dropouts and NH models without access to history.

point by following another lane than the reference lane.

Unsurprisingly, the models trained and evaluated without motion history (*-NH*) perform far worse than the models that had access to past observations. While the predicted trajectories are more diverse, the displacement errors are much higher. This confirms that motion history does provide crucial information for forecasting the future behavior of a TV. In contrast, the models trained with dropouts on the motion history (*-DO*) achieve competitive performance compared to the models that had access to it at all times. However, as we will demonstrate later, this does not improve generalization capabilities to perturbed scenes. As expected, the Constant-Acceleration model (CA) is a poor contestant with the highest displacement errors and off-road probabilities. Applying our Frenet wrapper to this simple heuristic significantly reduces the displacement errors. At the same time, the off-road probability (ORP) drops, as expected, to almost zero. The small remaining ORP is caused by samples where the predicted trajectory exceeds the map limits. The high displacement errors show that estimating the future motion of a TV requires complex reasoning about the scene context, even in the Frenet frame. We conclude that the proposed Frenet wrapper yields competitive performance compared to the Cartesian state-of-the-art baseline models in average scenarios.

4.5.2 Comparison on Perturbed Scenes

In the following, we analyze our results on the scene-attack benchmark (see Tab. 4.2) to evaluate the robustness of the models under map perturbations. For the original models, we observe that displacement errors and ORP increase drastically when perturbing the scenes. We observe an ORP increase from 0.7% to 63.9% and from 1.7% to 67.1% in the Double Turn perturbation for LaneGCN (Liang *et al.*, 2020) and for Multipath++ (Varadarajan *et al.*, 2022), respectively. This is consistent with what has been observed by (Bahari *et al.*, 2022). In contrast, *SD* models are much more robust, with ORP only increasing by 2.5

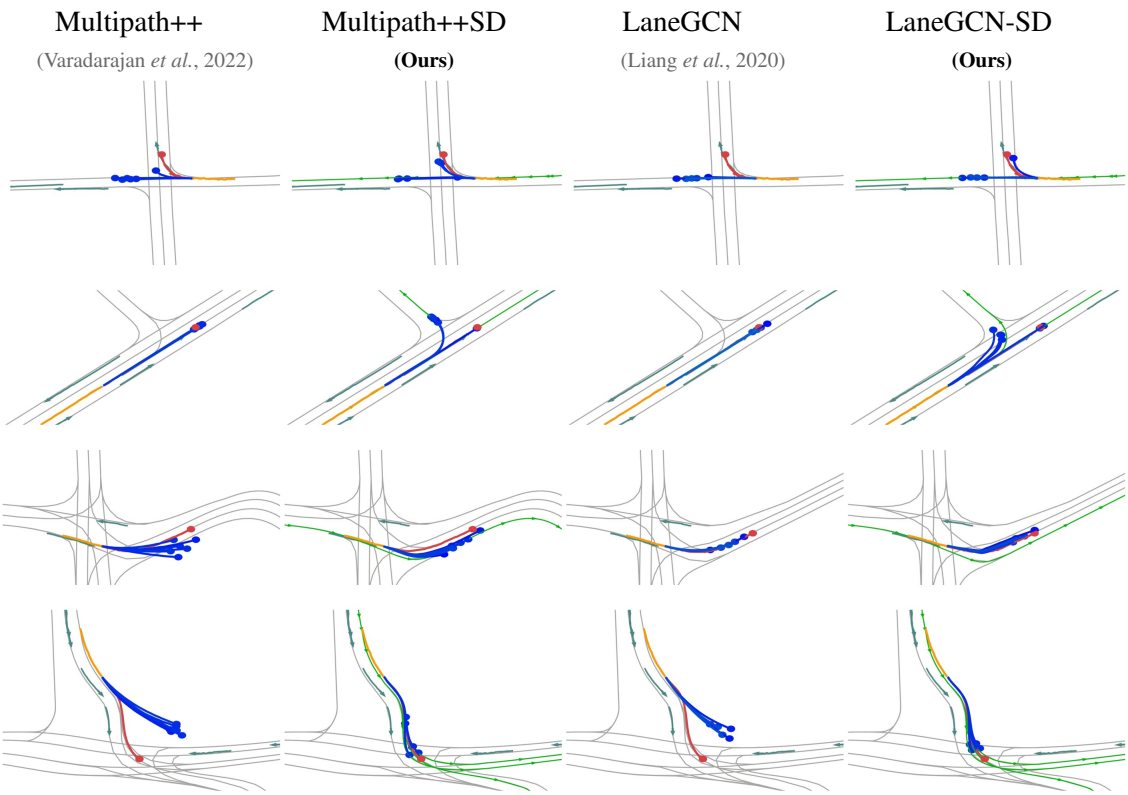


Figure 4.4: **Qualitative Results.** From top to bottom, two original and two perturbed scenes are shown. centerlines used by the *SD* models are shown in green, and state history in yellow. Predicted trajectories and ground truth are shown in blue and red, respectively. In the top example, the TV has just passed the branching point so that only the straight centerline is detected. The *SD* models are still able to predict the right-turn mode, significantly off the reference centerline. The second row demonstrates the increased diversity of *SD* model predictions. The third row shows deviations from the reference line. In the bottom example, the *SD* models adapt to a difficult perturbation, while the original models predict only off-road trajectories (Best viewed in color).

Model	prior / mode selection	$\hat{K}$	minADE ↓ [m]	minFDE ↓ [m]	ORP ↓ [%]	MR1 ↓ [%]	MIED ↑ [m]
Multipath++	-	6	0.772	1.399	1.7	**56.7**	2.908
Multipath++SD*	privileged	6	0.808	1.390	**1.2**	59.3	3.098
MultiPath++SD-UP	uniform / all	$N \cdot 6$	**0.767**	**1.274**	1.4	61.7	**3.473**
MultiPath++SD-UP	uniform	6	0.936	1.742	1.6	61.7	2.942
Multipath++SD-LS	lane-scoring	6	0.915	1.688	1.5	59.8	3.019
Multipath++SD-KM	clustering	6	0.833	1.470	1.3	65.3	3.518
Multipath++SD-GS	greedy-sampling	6	0.837	1.470	**1.2**	59.8	3.216
LaneGCN	-	6	**0.645**	**1.076**	**0.7**	**48.7**	3.800
LaneGCN-SD*	privileged	6	0.716	1.173	1.1	53.6	4.347
LaneGCN-SD-UP	uniform	6	0.777	1.335	1.3	55.4	4.091
LaneGCN-SD-UP	uniform / all	$N \cdot 6$	0.704	1.138	1.3	55.4	**4.678**
LaneGCN-SD-LS	lane-scoring	6	0.760	1.292	1.0	54.1	4.016
LaneGCN-SD-KM	clustering	6	0.759	1.315	1.4	65.1	4.703
LaneGCN-SD-GS	greedy-sampling	6	0.737	1.233	1.1	54.1	4.261

Table 4.3: **Methods to estimate the lane prior.**

percentage points for both models. Note that the perturbed scenarios are more challenging than the unperturbed ones, e.g. there are no straight driving scenarios, which explains the moderate increase in ORP. Additionally, the pseudo ground-truth used to evaluate the displacement errors causes uncertainty in calculating these metrics. Our results also show that neither Dropouts in training (*DO*) nor removing the motion history entirely from the features (*NH*) make the models more robust. Overall, the *SD* models outperform all the baselines by a large margin, resulting in a new State-of-the-Art on the scene-attack benchmark.

4.6 Ablation Study

In the following, we ablate different methods to estimate the prior $p(C^{(i)})$ over the centerlines. We start by introducing several methods for this task, ranging from a uniform prior assumption over a learning-based estimation to clustering based methods. Results are shown in Tab. 4.3.

4.6.1 Estimating the Lane Prior

Privileged Prior. Models denoted with an asterisk * only predict trajectories conditioned on the ground truth future centerlines, i.e. they have perfect centerlines scoring, and results serve as an upper bound. We find that this significantly reduces the gap between the models with $\hat{K} = N \cdot 6$ and the ones with $\hat{K} = 6$. Moreover, Multipath++SD* even outperforms the Cartesian Multipath++ (Varadarajan *et al.*, 2022) model. This emphasizes that an accurate estimation of the prior over the lanes significantly improves prediction

performance on the original scenes. Hence, we conclude that the reduced performance of the $\hat{K} = 6$ models without the perfect lane scoring is mostly caused by the naive assumption of a uniform prior over the lanes.

Uniform Prior. In contrast to the privileged prior models, models denoted with UP assume a uniform prior over the centerlines $C^{(i)}$, i.e. $p(C^{(i)}) = \mathcal{U}$. Then, we pick the $\hat{K}$ trajectories with the highest probabilities $p(t_i)$, which are obtained by marginalising the predicted conditioned probabilities $p(t_i|C^{(i)})$. We use the same uniform prior assumption in order to evaluate the miss-rate when selecting a variable number of trajectories $\hat{K} = N \cdot 6$.

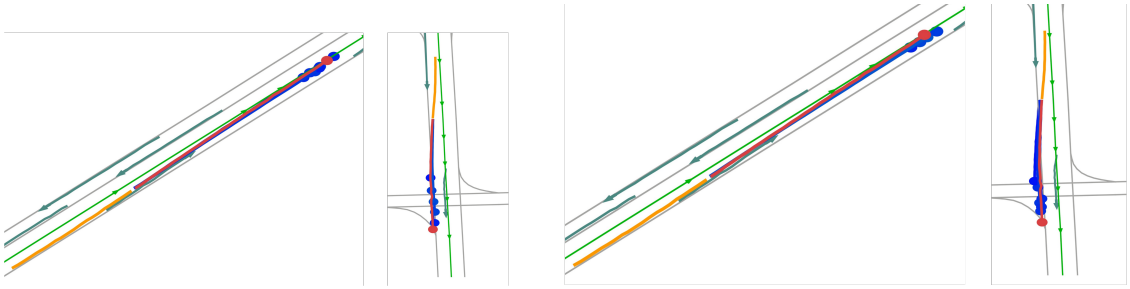
Lane Scoring Model. We employ a simple feedforward model that predicts a score for each centerline in the scene. A subsequent softmax layer across all centerlines in the scene yields the probability for each lane. The model inputs comprise the target vehicle's past trajectory as well as a sequence of poses along the centerline, given in Cartesian coordinates w.r.t. the current pose. Both are independently processed by a single linear layer and then concatenated to form a feature vector. Afterwards, an MLP with two hidden layers, each comprising 512 neurons, regresses the score for the given centerline. The lane scoring model is trained for ten epochs at a learning rate of $1e-4$ with a batch size of 128. We use the cross-entropy loss as training objective. We denote the models that are evaluated with the lane scoring prior with the suffix *LS*.

Clustering. As an alternative to estimating the prior, we employ a K-means clustering of the $N \cdot K$ trajectories to obtain a set of K output trajectories (Deo *et al.*, 2022). The estimated probability of each trajectory is obtained from the inverse rank of the cluster. We denote this ablation with the suffix *KM*.

Greedy Sampling. Similar to (Gilles *et al.*, 2021a), the set of K output trajectories can be selected greedily. This extends the idea of the lane scoring model with a non-maximum suppression. First, the lane scoring model is inferred to obtain the estimated probability of each trajectory. Next, a greedy algorithm selects the most probable trajectory and removes all trajectories from the candidate set whose endpoint is within a radius of 1.0m of the selected one. This is repeated until $K = 6$ trajectories are selected. We denote this model with the suffix *GS*.

4.6.2 Results

We find that selecting a set of $\hat{K} = 6$ trajectories with a uniform prior results ($-UP$) in the highest displacement errors. As the ablation that leverages a privileged prior (*) performs substantially better, we conclude that the impaired performance results from a suboptimal selection of the trajectories. Consequently, leveraging the lane-scoring model (*LS*) to estimate the prior over the lanes results in decreased displacement errors. Moreover, the miss-rate closely approaches the level of privileged prior ablation. Combining this with non-maxima-suppression, as it is done by our greedy sampling method (*GS*), results in the lowest displacement errors among the Frenet models (*SD*) while maintaining the low miss rate. Besides, the diversity (MIED) is higher than for the lane-scoring models, which select the trajectories with the highest estimated probability.



(a) LaneGCN-SD (b) Multipath++SD

Figure 4.5: The left images in both subfigures shows how the models predict a lane change onto an adjacent lane. In the right image of each subfigure, the target vehicle already started to change to the neighboring lanes. Both models are able to forecast a feasible completion for that lane change, even if they are conditioned on the centerline that the target vehicle is about to leave.

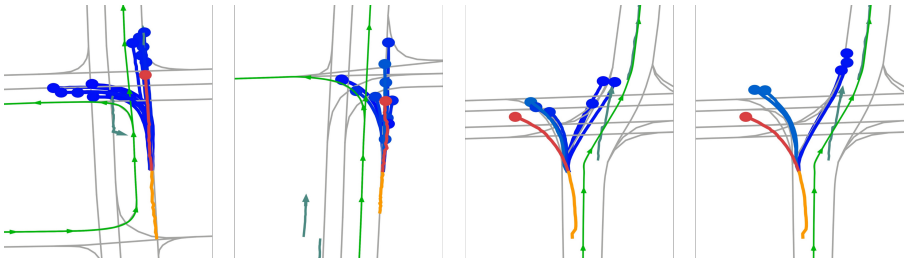


Figure 4.6: The two left columns show how the Multipath++SD model (leftmost) and the LaneGCN-SD model (second from left) predict an illegal left turn if they are conditioned on the neighboring centerline. The two right columns show how the models are able to anticipate a left turn because the vehicle started turning to the left, even though they are conditioned on a centerline going straight. It can be seen how this significant deviation from the centerline results in an offroad prediction in the case of the LaneGCN-SD model (rightmost).

4.7 Discussion

The qualitative results shown in Fig. 4.4 demonstrate how the Frenet models generalize seamlessly to the perturbed scenes, whereas the Cartesian models fail to adapt. Furthermore, we find that the remaining off-road predictions of the Frenet models are mostly caused by corner cases in the identification of relevant centerlines. For instance, in cases where the TV is positioned directly behind a lane branching point, assigning it to a single lane results in not detecting the other lane as a relevant centerline. We find that the model partially compensates for this, by also following other lanes than the reference lane of the coordinate frame. Moreover, Fig. 4.5 and Fig. 4.6 demonstrate the prediction of lane changes and non-compliant behavior, respectively. These predictions can be induced by conditioning the model on a neighboring centerline. We also observe that a lane change or a turn that has already been initiated is completed, even if the model is conditioned on the centerline that the target vehicle is about to leave. However, we observe that there are cases in which some waypoints of the resulting trajectories are slightly off-road. Nonetheless, the results clearly show that the models do not blindly follow the centerline of the Frenet frame. On the contrary, as can be seen in the qualitative results, the models are able to deviate significantly from it, indicating a thorough understanding of the scene.

Conclusion. In this chapter, we presented a general Frenet transform wrapper compatible with state-of-the-art prediction models and demonstrated its effectiveness on two different state-of-the-art prediction models. We found that representing the input and output of prediction models in the Frenet frame improves generalization to challenging scenarios by a margin. Hence, we were able to reduce predicted off-road probability by over 90% on the scene-attack benchmark. Moreover, our wrapper comes with the advantage of more diverse predictions, increasing MIED score (a measure for the diversity of predicted trajectories) by 20%, while prediction accuracy is on par or better than for the representation in the Cartesian frame.

Limitations. In the case of a fixed number of predicted trajectories, the robustness of our method comes at the cost of slightly reduced prediction accuracy. Moreover, our results indicate that a better estimation of the prior over the relevant lane sequences can enhance the predicted mode probabilities. Finally, our Frenet wrapper was designed with map-based predictors in mind. How these models can be applied in unstructured environments (e.g., parking lots) remains unclear.

Chapter 5

From Prediction to Planning

5.1 Introduction

The previous chapters covered the trajectory prediction task and research questions surrounding it. In the following, we will delve into the downstream planning task. The anatomy of both tasks presented in chapter 2 already indicated that learning-based prediction and planning algorithms exhibit large similarities regarding input and output representations. Both aim to regress the trajectory a target vehicle (TV) will follow, based on a given scene context. However, the fields differ in key aspects, namely whether the intention is assumed to be known and the requirements placed on the output trajectory(s). In particular, in contrast to planning, prediction methods forecast the motion of observed agents solely based on current and past observations, *i.e.,* without knowing their intention. To account for the unknown intention, prediction methods commonly regress multiple trajectories that an agent could follow. Consequently, the diversity of the predicted trajectories, *i.e.,* the number of captured potential intentions, plays an important role. In contrast, in the planning task, the ego vehicle (EV)'s intention is presumed to be known, making diversity at the intention level obsolete. Moreover, as the EV is assumed to follow the trajectory produced by a planning algorithm, collision avoidance, comfort, and kinematic feasibility are fundamental requirements for a planner's output.

This makes collecting datasets and learning prediction models through supervised learning straightforward. Promoted by large open-source datasets with associated benchmarks (Ettinger *et al.*, 2021; Caesar *et al.*, 2020; Wilson *et al.*, 2023) the field of vehicle trajectory prediction has seen tremendous progress recently. At the same time, learning a planner from these datasets is difficult, because observation alone without knowing the intention is not sufficient to unambiguously understand and imitate an expert's decision (Codevilla *et al.*, 2018a). Hence, carrying over advances from prediction to planning is a challenging task.

In this chapter, our goal is to incorporate intention into a state-of-the-art predictor, converting it into an effective planner. Thus, we build on the existing vehicle trajectory prediction model PGP (Deo *et al.*, 2022) and introduce a novel method for goal conditioning that precludes plans going off-route from being considered. At the same time, the model retains its diversity at the motion level, such that it doesn't blindly follow a goal

without reasoning about safety and comfort.

Our main contribution is as follows: We introduce a novel method for goal-conditioning, which allows us to leverage advances in the field of prediction for the task of planning. We leverage our method to convert the unconditioned PGP prediction model (Deo *et al.*, 2022) into a goal-conditioned planner. We evaluate this planner in open-loop and closed-loop simulation on the nuPlan open-source dataset (Caesar *et al.*, 2021) and observe that this simple yet effective modification alone can significantly reduce the gap towards state-of-the-art planning performance.

The remainder of this chapter is structured as follows: First, we discuss related work and recent advances in prediction and planning. Subsequently, in Section 5.3 we introduce a novel approach for goal-conditioned planning, which is evaluated in Section 5.4. Finally, Section 5.5 draws conclusions and gives an outlook on promising future research directions.

5.2 Related Work

Prediction and Planning for Automated Driving. The planning task aims to find a single trajectory that is optimal in terms of driving comfort, safety, efficiency, progress along the route, etc. Besides rule-based approaches, which can suffer from poor generalization to new scenarios and need large engineering efforts, learning-based approaches have gained traction over recent years. Pioneering work in the 1980s (Pomerleau, 1988) demonstrated lane following on public roads with a shallow dense neural network using Behavior Cloning (BC). A recent line of work leverages neural networks in optimization-based planning by learning cost functions from data (Zeng *et al.*, 2019; Wei *et al.*, 2021; Sadat *et al.*, 2020; Casas *et al.*, 2021). The trajectory prediction task is closely related to the task of planning, often using similar input and output representations. A large body of literature exists, with new models showing ever-increasing performance on public benchmarks (Lu *et al.*, 2022; Varadarajan *et al.*, 2022; Liang *et al.*, 2020; Nayakanti *et al.*, 2022; Ye *et al.*, 2022, 2021; Shi *et al.*, 2022b). Despite their similarities, prediction and planning are often tackled as separate tasks using different model architectures. This makes it difficult for the field of planning to benefit from advances in the field of prediction. In this chapter, we show how an existing state-of-the-art prediction model can be repurposed for the task of planning, directly carrying over advances from one task to the other.

From Grids to Graphs. Representing the environment in the form of rasterized grids in combination with a convolutional neural network backbone has been a very popular approach for trajectory prediction (Cui *et al.*, 2019; Gilles *et al.*, 2021a; Luo *et al.*, 2018; Chai *et al.*, 2019; Phan-Minh *et al.*, 2020). In recent years a new generation of models emerged that relies on graph-based representations for the map (Varadarajan *et al.*, 2022; Gilles *et al.*, 2022; Liang *et al.*, 2020; Deo *et al.*, 2022) and leverages graph neural networks or attention blocks to model interactions (Gilles *et al.*, 2021b; Yuan *et al.*, 2021; Khandelwal *et al.*, 2020; Casas *et al.*, 2020c; Gao *et al.*, 2020; Knittel

et al., 2022; Nayakanti *et al.*, 2022). We refer the reader to (Liu *et al.*, 2021b) for a comprehensive survey. This innovation has not yet completely carried over to the planning task, with (Scheel *et al.*, 2022; Wang *et al.*, 2022a) being notable exceptions. By repurposing a state-of-the-art prediction model for the task of planning, we are able to directly transfer recent advances, including model architectures and input representations, from prediction to planning.

Goal Conditioning. A key challenge to repurposing a prediction model for planning is how to condition the diverse, multi-modal outputs on a given intention, *i.e.* the navigation goal. The two predominant alternatives are to either switch between specialized submodels depending on a high-level command or add additional model inputs (Codevilla *et al.*, 2018a). The former is an effective way to reduce the imbalance in the training set and has been successfully applied in (Casas *et al.*, 2021). However, as the number of high-level commands and their definition have to be fixed in advance, this comes at the cost of flexibility. Hence, the latter approach, *i.e.* passing the intention as an additional input to the model, has been applied more broadly (Chitta *et al.*, 2021; Prakash *et al.*, 2021; Shao *et al.*, 2023b; Chen and Krähenbühl, 2022). As already observed in (Codevilla *et al.*, 2018a) these models are only implicitly conditioned on the intention, with the risk that the route is not always followed. Our approach circumvents this risk by conditioning directly on the intention level. Another way to encourage goal-directed planning is the addition of a cost term that favors progress along the route (Sadat *et al.*, 2020). The optimizer then has to find a balance between progress and other objectives, such as safety and comfort, possibly sacrificing one for the other.

Goal-Conditioned Prediction as Planning. Two-step prediction approaches separate the level of intended behavior from the level of exact motion trajectory instead of directly regressing the latter. Various representations for the behavior level have been proposed. (Wang *et al.*, 2022a) decodes trajectories conditioned on a target lane, whereas (Zhao *et al.*, 2021b; Gu *et al.*, 2021) use the same paradigm but condition on concrete spatial locations or sparsely sampled trajectories (Lu *et al.*, 2022). We build our work on (Deo *et al.*, 2022), which conditions predictions on a coarse path represented by a traversal of the lane graph. Since this path does not include any temporal information, such as the velocity profile or exact spatial locations, it represents a behavior-level plan that can be followed with multiple motion trajectories. However, the approach from (Deo *et al.*, 2022) does not consider routing in the traversal selection. We show that this separation of intention and motion level opens up an opportunity to define a new method of goal conditioning. In particular, we demonstrate that by goal-conditioning of the traversals, the EV can only consider route-compliant plans while still being able and required to reason about safety in order to plan the exact trajectory.

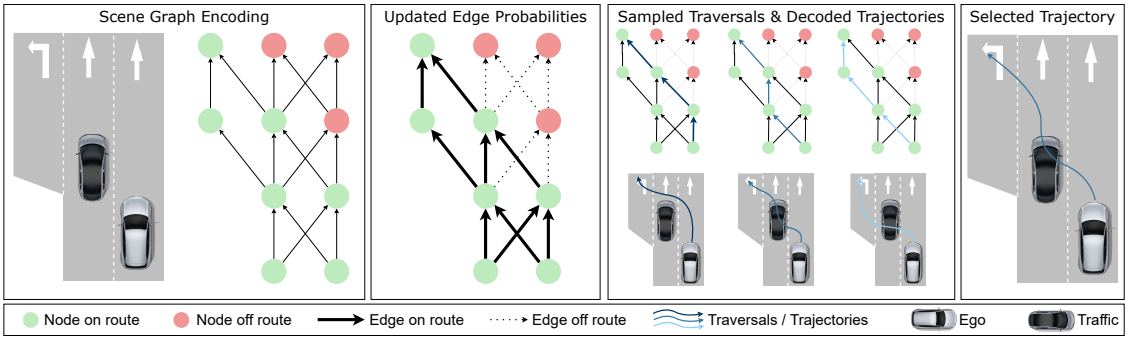


Figure 5.1: **Method Overview.** We manipulate the edge probabilities of a graph-based scene representation. Route-conditioned trajectories are then decoded from the manipulated graph, and the most likely one is selected as a plan for the ego vehicle (EV).

5.3 Method Description

5.3.1 Graph-Based Prediction Model

We give a brief recap of the unconditioned PGP prediction model (Deo *et al.*, 2022), which we convert into a goal-directed planner by goal-conditioning at the level of intended behavior. An overview of the PGP model architecture can be found in Fig. 5.2. We refer the reader to (Deo *et al.*, 2022) for a thorough description of the graph representation and details on the prediction model.

Encoder. The encoder builds a graph representation of the environment centered around a road graph G consisting of nodes V and directed edges E, i.e., $G(V,E)$. The graph nodes V represent lane centerlines snippets. Therefore, lanes are divided into snippets of similar length and discretized into a polyline. Hence, the node features X_v are a sequence of N_v poses describing the centerline snippet v, i.e. $X_v = \{x_i^{(v)}, y_i^{(v)}, \theta_i^{(v)}, S_i^{(v)}, C_i^{(v)}\}_{i=0}^{N_v}$, where $x_i^{(v)}, y_i^{(v)}, \theta_i^{(v)}$ are the position and orientation given in the EV's local cartesian coordinate frame. Binary flags $S_i^{(v)}$ and $C_i^{(v)}$ indicate stop lines and crosswalks respectively. Two types of directed edges $E = E_{\text{succ}} \cup E_{\text{prox}}$ model allowed transitions between the nodes: E_{succ} for successor nodes along the lane in the direction of traffic flow and E_{prox} for proximal nodes on neighboring lanes.

As defined in section 2.2, for each agent a in the scene its history over the observation horizon $t = t_0 - t_h, .., t_0$ is represented as a trajectory $X_a = \{s_t^{(a)}\}_{t=t_0-t_h}^{t_0}$. Each state $s_t^{(a)}$ is defined as $s_t^{(a)} = [x_t^{(a)}, y_t^{(a)}, v_t^{(a)}, a_t^{(a)}, \omega_t^{(a)}, I^{(a)}]$, where $x_t^{(a)}, y_t^{(a)}$ are the position given in the EV's local cartesian coordinate frame and $v_t^{(a)}$, $a_t^{(a)}$, $\omega_t^{(a)}$ denote the agent's speed, acceleration, and yaw rate, respectively. $I^{(a)}$ is an indicator for agent class (vehicle or pedestrian).

Polylines X_v, agents' histories X_a, and the EV history X_{EV} are all encoded using GRUs, yielding encodings h_{node}^v, h_{agent}^a, and h_{EV}. Agent-to-node attention allows the lane nodes

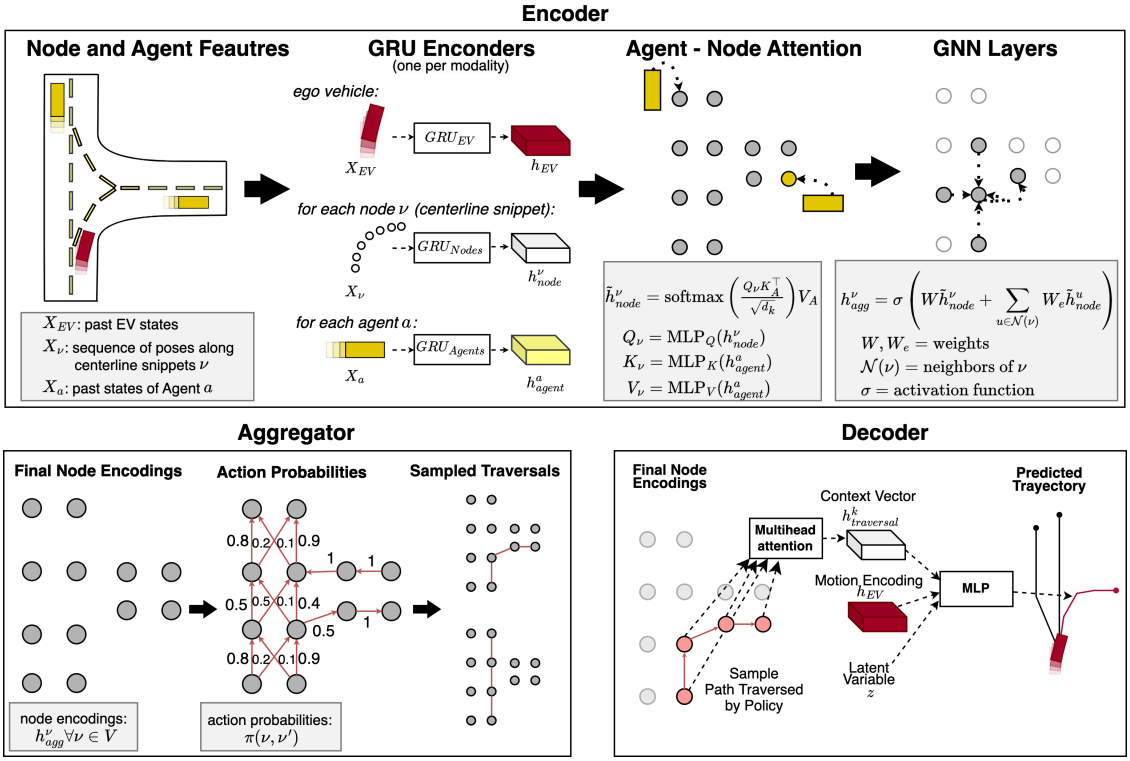


Figure 5.2: **Overview of the PGP Model.** Adapted from (Deo *et al.*, 2022). The model consists of three modules. First, the encoder takes the history of surrounding agents as well as the EV motion and map information as input and encodes with Gated Recurrent Units (GRUs). Agent-Node Attention and GNN layers generate a road graph containing real-time traffic information. Subsequently, the policy header predicts probabilities for the outgoing edges of each graph node. Sampling these probabilities yields traversals, that describe a likely future behavior. Finally, the latent-variable decoder regresses the exact motion trajectory based on the traversals.

to accumulate traffic information before graph neural network GNN layers are applied to the road graph. The full encoding is given by $h^v_{\text{agg}} \, \forall v \in V$.

Graph Traversals. The *aggregator* module determines transition probabilities for all edges and then samples traversals through the graph. Thus, for each node $v \in V$, the probabilities for all outgoing edges (v, v') are obtained from

$$\text{score}(v, v') = \text{MLP}(\phi(h^v_{\text{agg}}, h^{v'}_{\text{agg}}, h_{\text{EV}})) \tag{5.1}$$

$$\pi(v, v') = \text{softmax}\left(\text{score}(v, v') \,|\, (v, v') \in E\right), \tag{5.2}$$

where ϕ denotes concatenation. Next, graph traversals are obtained by assigning the EV to the closest node, then iteratively sampling an outgoing edge, and assigning the EV to the next node along this edge. To allow traversals of arbitrary length (up to a maximum number of T nodes), terminal edges are added to each node. At inference time, K traversals $\{v^k_n\}^T_{n=0}$, $k = 1, .., K$ are sampled.

Trajectory Decoder. The traversal encoding is

$$h^k_{\text{traversal}} = \phi(h^v_{\text{agg}} \, \forall v \in \{v^k_n\}^T_{n=0}). \tag{5.3}$$

A latent variable model then decodes the predicted trajectory

$$o^k = \text{MLP}(\phi(h^k_{\text{traversal}}, h_{\text{EV}}, z)) \tag{5.4}$$

from a given traversal and a randomly sampled noise vector z. In practice, the number of sampled traversals K is large, and a fixed number of output trajectories is obtained after k-means clustering. Each trajectory is assigned a probability based on the respective cluster's rank.

5.3.2 Route-Conditioned Traversals

We aim to ensure goal conditioning of the trajectory output at the behavior level. To this end, we manipulate the edge probabilities obtained by the prediction model so that traversals that follow the intended route become more likely. An overview of our method is depicted in Fig. 5.1. All nodes that lie on a path between the EV's current position and the navigation goal are considered "on route", and the corresponding edges form the set E_{route}.

We propose two alternative methods for conditioning road-graph traversals. First, we increase edge probabilities that stay on the route. Second, we set probabilities of edges that deviate from the route to zero. To enhance probabilities of edges that stay on the route, we add an additive bonus β to the regressed score

$$\pi(u.v) = \begin{cases} \text{softmax}(\text{score}(v, v') + \beta) & (v, v') \in E_{\text{route}} \\ \text{softmax}(\text{score}(v, v')) & (v, v') \notin E_{\text{route}} \end{cases} \tag{5.5}$$

The magnitude of the bonus β is identical for all edges and is a learnable model parameter. We call this setup *soft-mask* goal conditioning.

Alternatively, we apply a *hard mask* by setting the probabilities of edges that do not follow the route to zero. This provides a guarantee that all traversals comply with the navigation goal. The updated probabilities are given by

$$\pi(u.v) = \begin{cases} \text{softmax}(\text{score}(v, v')) & (v, v') \in E_{\text{route}} \\ 0 & (v, v') \notin E_{\text{route}} \end{cases} \tag{5.6}$$

We can apply the hard mask at inference time only (subsequently referred to as *Goal-Conditioned PGP (GC-PGP)*), or already during training (termed *hard mask*). Applying the mask only at inference time is also beneficial because the model still learns the entire multi-modal distribution, and no re-training is necessary.

5.3.3 Trajectory Selection

For a prediction model, it is a natural choice to output multiple trajectories per agent in order to account for the multi-modality induced by the unknown intention. In contrast, for the planning task, the model has to commit to a single trajectory. Similar to (Pini *et al.*, 2022) we apply a simple heuristic and select the trajectory with the highest probability.

Note that this choice does not necessarily correspond to the predominant road graph traversal (due to the clustering step).

5.4 Experiments

5.4.1 Dataset and Evaluation Framework

We employ the nuPlan dataset and framework (Caesar *et al.*, 2021) for model training and evaluation. The dataset includes diverse scenarios, e.g., making turns, yielding to pedestrians, stopping at intersections, traversing intersections, and driving at different speeds. The full nuPlan train split consists of more than 30 million scenarios totaling over 1,300 hours of real-world driving. For training, we use all of the available 70 scenario types at a maximum of 4,000 scenarios per type. For some types, less than 4,000 scenarios are available, resulting in approx. 150,000 training scenarios.

The nuPlan entire test split consists of 4,539 scenarios. We noticed annotation errors in some of these samples, which resulted in relevant lanes being labeled as off-route and vice versa. Since this is particularly harmful in the context of target conditioning, we remove these samples from the test set. By sampling a maximum of 30 scenarios for each available scenario type and removing the ones that are compromised by labeling errors, we obtain a test set consisting of 1,363 diverse driving scenarios.

In addition, we report results on a subset that focuses on scenarios where knowing the navigation goal is crucial. We limit this reduced test set to the following four scenario types at 100 scenarios each: traversing intersection, starting unprotected non-cross turn, starting protected cross turn, and starting right turn. After removing the compromised samples, this intersection test split consists of 323 scenarios.

5.4.2 Metrics

Besides the aggregated driving score defined by the nuPlan framework, we also report average displacement error (ADE), final displacement error (FDE), and miss rate (MR) for the open-loop evaluations. MR is defined by the final waypoint of the 8-second planning horizon deviating more than 16 meters from the ground truth. For the closed-loop simulations, we evaluate progress, driveable area compliance, and collision avoidance. Progress is given by the fraction of the traveled distance along the ground truth path. Driveable area compliance describes the share of frames where the EV's entire bounding box is on the driveable surface. Besides, we report the fraction of successful scenarios in terms of at-fault collisions. The EV is considered to be at fault for a collision if it collides with a stationary agent or if the collision occurs at its front or sides. We train and evaluate each model twice and report the mean and standard deviation for all metrics.

5.4.3 Baselines and Ablations

Intelligent Driver Model baseline. IDM (Treiber *et al.*, 2000) is a heuristic model that keeps a safe distance from the vehicle ahead. Its extension MOBIL (Kesting *et al.*, 2007) also handles lane changes. We use the implementation provided by the nuPlan framework (Caesar *et al.*, 2021).

Urban Driver baseline:. We compare our model to a state-of-the-art planning approach, which has been made available together with the nuPlan framework. Urban Driver (Scheel *et al.*, 2022) uses PointNet layers to encode the EV's motion as well as observed agents' history and map elements. Subsequently, the information is fused using multi-head attention. Then, a trajectory for the EV is decoded from the aggregated encoding. In contrast to our method, for this baseline, the route information is encoded into the map features. The model is fairly large (2.2M parameters), so we adapt its complexity for improved comparability to PGP (150k parameters). We reduce the encoding size from 256 to 128 and the number of subgraph layers from 3 to 2, resulting in 520k parameters (*UD-520k*). Similarly, the *UD-150k* model uses an encoding size of 64, but retains all 3 subgraph layers, totalling 150k parameters.

Filter on route baseline. The multi-modal output of PGP is post-conditioned on the route: To this end, we only keep trajectories whose endpoint is within 5 meters of the nearest lane center on the route; among those, we pick the trajectory with the largest probability. If no trajectory is within the distance threshold, we pick the one closest to the route.

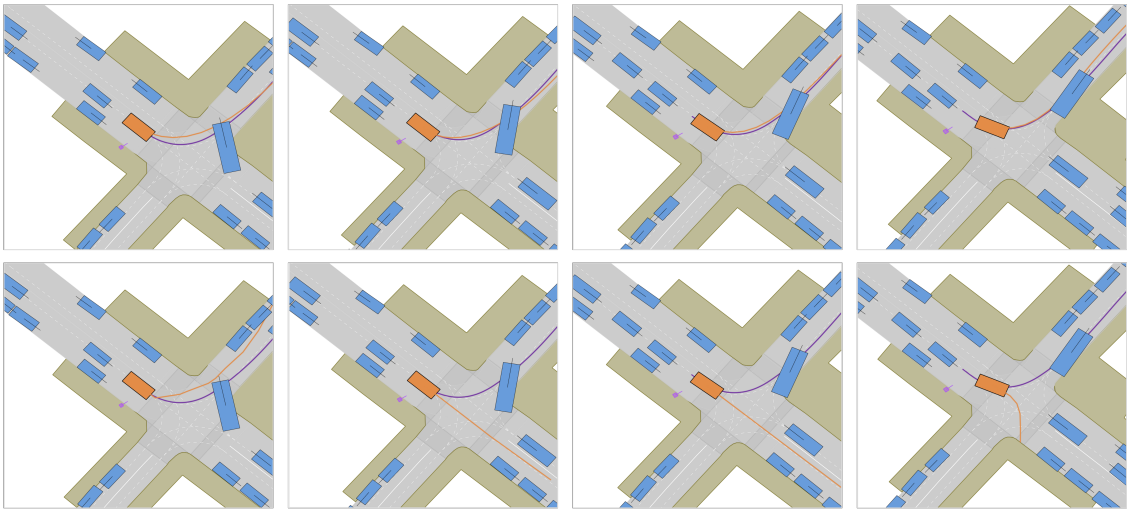


Figure 5.3: **Qualitative results.** Left to right depicts the first 4 seconds of the EV making a left turn. Top: GC-PGP, Bottom: *node features* ablation. The ego vehicle and its planned trajectory are shown in orange, the expert's path in purple. Surrounding vehicles and pedestrians are blue and green respectively.

Node features ablation. An alternative to explicitly conditioning a model on the navigation goal is to extend the model's input features (Codevilla *et al.*, 2018a). Hence, we add a binary value indicating whether the node is part of the route to each node in the lane graph.

Soft mask ablation. The transition probabilities of edges that keep the traversal on route receive a (learnable) bonus, see Eq. (5.5).

Hard mask ablation. Unlike our GC-PGP model, edges that leave the route are removed, also at train time (see Eq. (5.6)). This way, both the aggregator and trajectory decoder models are restricted to goal-reaching traversals.

5.4.4 Open-Loop Simulation

We report in Tab. 5.1 open-loop imitation performance on the full, diverse test split. This dataset is dominated by scenarios where route information is not required, such as being stationary at traffic lights or simple lane following. We observe that all goal-conditioning variants still outperform the (undirected) PGP baseline in all metrics, closely approaching the strong Urban Driver baseline. Interestingly, the IDM/MOBIL model is a poor contestant w.r.t. imitation metrics: its behavior may have desirable properties, but it is not particularly human-like on nuPlan scenarios.

To demonstrate the effectiveness of our approach, we also evaluate on a reduced test set as described in Sec. 5.4.1. This dataset is focused on non-trivial intersection scenarios

Model	Score ↑	ADE [m] ↓	FDE [m] ↓	MR ↓
IDM/MOBIL	0.33	7.98	12.8	0.42
Urban Driver	**0.84 ± 0.01**	**1.42 ± 0.00**	**2.99 ± 0.07**	**0.06 ± 0.00**
UD-520k	0.83 ± 0.00	1.49 ± 0.03	3.18 ± 0.08	0.06 ± 0.02
UD-150k	0.80 ± 0.01	1.62 ± 0.02	3.39 ± 0.00	0.08 ± 0.00
PGP	0.69 ± 0.02	1.82 ± 0.06	4.21 ± 0.14	0.14 ± 0.00
GC-PGP **(Ours)**	0.76 ± 0.01	1.59 ± 0.03	3.65 ± 0.09	0.12 ± 0.01
Filter on route	0.73 ± 0.01	1.66 ± 0.01	3.84 ± 0.03	0.13 ± 0.00
Node features	0.73 ± 0.01	1.70 ± 0.02	3.91 ± 0.02	0.13 ± 0.00
Soft mask	0.71 ± 0.00	1.75 ± 0.00	4.02 ± 0.05	0.14 ± 0.01
Hard mask	0.62 ± 0.04	2.23 ± 0.27	4.82 ± 0.99	0.17 ± 0.05

Table 5.1: **Entire nuPlan test split.** Results of the open-loop evaluation.

Model	Score ↑	ADE [m] ↓	FDE [m] ↓	MR ↓
IDM/MOBIL	0.19	4.89	5.81	0.20
Urban Driver	**0.83 ± 0.00**	**1.66 ± 0.02**	**3.36 ± 0.03**	**0.05 ± 0.00**
UD-520k	0.81 ± 0.00	1.80 ± 0.04	3.73 ± 0.09	0.07 ± 0.00
UD-150k	0.77 ± 0.00	1.96 ± 0.04	4.03 ± 0.08	0.08 ± 0.00
PGP	0.56 ± 0.03	2.48 ± 0.08	5.67 ± 0.20	0.19 ± 0.01
GC-PGP **(Ours)**	0.74 ± 0.01	1.80 ± 0.04	4.03 ± 0.11	0.10 ± 0.00
Filter on route	0.68 ± 0.01	1.98 ± 0.01	4.50 ± 0.04	0.14 ± 0.00
Node features	0.67 ± 0.02	2.15 ± 0.07	4.81 ± 0.12	0.14 ± 0.01
Soft mask	0.60 ± 0.00	2.41 ± 0.00	5.43 ± 0.05	0.19 ± 0.01
Hard mask	0.67 ± 0.04	2.20 ± 0.15	4.92 ± 0.29	0.16 ± 0.03

Table 5.2: **Intersection Scenarios.** Results of the open-loop evaluation

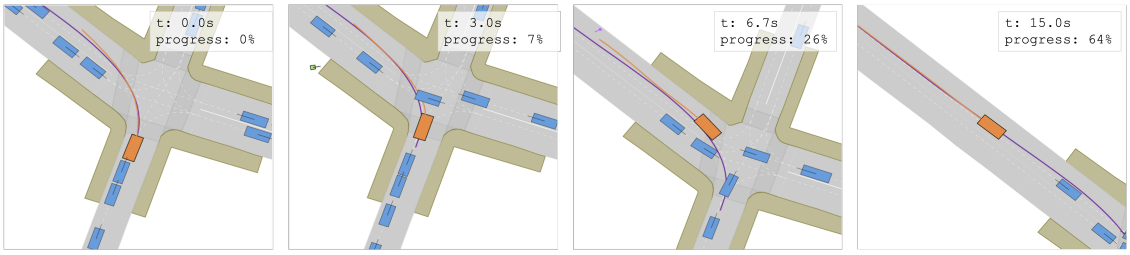


Figure 5.4: **GC-PGP making a Turn in closed-loop simulation**. The EV enters the intersection after 3 seconds, having traveled 7% of the expert's total traveled distance, and leaves the intersection after 6.7 seconds with 26% progress. After 15 seconds of simulation, it covers 64% of the expert's path. (Best viewed in color.)

where the navigation goal is important to the driving behavior. Results are reported in Tab. 5.2 and confirm our previous observations. Our goal-conditioned model (*GC-PGP*) clearly outperforms the PGP prediction baseline and catches up to the Urban Driver planner baseline with similar size (*UD-150k*) in overall score and displacement error. Moreover, it outperforms all alternative goal-conditioning methods we evaluated. We observe that the *node features* and *soft mask* models only show mild improvements over PGP. We speculate that they do not always strictly adhere to the route information, a claim that is supported by qualitative results in Fig. 5.3. In contrast, the masking in our GC-PGP model prevents taking plans off the route into consideration. However, we note that applying this hard mask also at train time to focus on goal-directed traversals only compromises performance. This model is much less robust against route labeling errors as described in Sec. 5.4, and we hypothesize about a connection between the resulting faulty mask and observed instabilities during training.

5.4.5 Temporal Plan Stability

Model	**TPI [m]** $\downarrow$
IDM/MOBIL	**0.58**
Urban Driver	1.51 ± 0.00
UD-520k	1.78 ± 0.02
UD-150k	1.84 ± 0.14
PGP	3.65 ± 0.15
GC-PGP **(Ours)**	2.80 ± 0.00

Table 5.3: **Temporal plan instability on intersection scenarios**

The deviation between planned trajectories for consecutive timesteps is a measure of the robustness of the planner. A low value means the plans are consistent over time,

indicating robustness and a comfortable driving experience. We define temporal plan instability (TPI)

$$\text{tpi}(\tau) = \| (x,y)_T^{@\tau} - (x,y)_{T-1}^{@\tau+1} \|_2 \tag{5.7}$$

the distance between (adjusted) trajectory end-points for consecutive timesteps ($@\tau$, $@\tau+1$) and report results in Tab. 5.3.

The IDM/MOBIL baseline produces the most time-consistent plans and can serve as a lower limit, which is unsurprising given its underlying simple heuristics. PGP is the least time-consistent model. As a prediction model, it has been designed to produce diverse trajectories, both w.r.t. the route taken and different velocity profiles. Toggling between these modes naturally leads to large plan instability. Conditioning on a navigation goal (GC-PGP) removes the main source of multi-modality, but diversity w.r.t. velocity remains as a cause for instability. Thus, GC-PGP yields more stable plans than PGP but retains a higher degree of toggling than Urban Driver, an inherently unimodal planning model. This indicates that the naïve way we select a trajectory among the multi-modal proposals may not be ideal. Consequently, planning performance could be further improved by applying a cost function that balances different aspects of driving, such as safety, comfort, progress, etc., as in (Sadat *et al.*, 2020; Casas *et al.*, 2021).

5.4.6 Closed-Loop Simulation

Model	Score ↑	Progress ↑	Driv. Area ↑	Col.-Free ↑
IDM/MOBIL	**0.75**	**0.81**	**0.96**	0.90
Urban Driver	0.58 ± 0.06	0.76 ± 0.11	0.89 ± 0.02	0.85 ± 0.01
UD-520k	0.50 ± 0.08	0.71 ± 0.06	0.90 ± 0.02	0.84 ± 0.02
UD-150k	0.45 ± 0.02	0.74 ± 0.08	0.81 ± 0.01	0.79 ± 0.03
PGP	0.37 ± 0.05	0.36 ± 0.07	0.88 ± 0.04	0.89 ± 0.03
GC-PGP (ours)	0.46 ± 0.03	0.47 ± 0.08	0.88 ± 0.01	0.88 ± 0.04
Filter on route	0.43 ± 0.02	0.43 ± 0.09	0.87 ± 0.04	0.87 ± 0.01
Node features	0.37 ± 0.00	0.37 ± 0.03	0.86 ± 0.05	0.87 ± 0.03
Soft mask	0.29 ± 0.02	0.30 ± 0.06	0.91 ± 0.00	$\mathbf{0.92 \pm 0.02}$
Hard mask	0.43 ± 0.04	0.43 ± 0.06	0.84 ± 0.01	0.84 ± 0.01

Table 5.4: **Intersection Scenarios.** Results of the closed-loop evaluation.

We also evaluate all models in the log-playback closed-loop simulation environment provided by the nuPlan framework. The same reduced test set as for the open-loop evaluation is used (cf. Sec. 5.4.1). Results are presented in Tab. 5.4. It is striking that the hand-crafted IDM/MOBIL model achieves the best overall performance by a large margin but is still not perfectly safe in terms of drivable area compliance and caused collisions. This completes the picture we gained from the open-loop evaluation (Sec. 5.4.4): IDM/MOBIL is a strong "classical" baseline, just not in terms of expert imitation.

Both PGP and GC-PGP are on par with Urban Driver regarding drivable area compliance and even outperform w.r.t. caused collisions. Although GC-PGP improves upon (the undirected) PGP in terms of progress along the route, with a value of 0.47 it still falls behind Urban Driver. See Fig. 5.4 to put this value into perspective: At some point well behind the crossing, GC-PGP fails to progress along the lane fast enough. We conclude that an average progress of only 0.47 is, on average, sufficient to master the crossing but indicates shortcomings in the subsequent lane following task, which exceeds the scope of our method.

For the alternative goal-conditioning approaches (lower part of Tab. 5.4), we do not observe significant deviations from their PGP and GC-PGP counterparts w.r.t. drivable area compliance and caused collisions. Instead, we do find that they all achieve less progress than GC-PGP, the *soft mask* variant being even worse than the PGP baseline. These results confirm GC-PGP as the most performant variant to condition PGP on a navigation goal.

5.5 Conclusions

In this chapter, we presented a novel way to achieve goal conditioning on a graph-based map representation that precludes behavior-level plans going off-route from being considered. We demonstrated the effectiveness of our approach in a comprehensive benchmark by repurposing a state-of-the-art prediction model for the planning task and comparing it against rule-based and data-based models, thereby highlighting that goal-conditioned prediction models can serve as strong baselines in planning benchmarks and as effective starting points for novel planning methods.

Chapter 6

Planning in Realistic Urban Environments

6.1 Introduction

As demonstrated in the previous chapter, learning-based planning baselines can easily be obtained from powerful trajectory prediction methods. Equipped with the method presented in the previous chapter (Hallgarten *et al.*, 2023a), we now perform a rigorous empirical analysis on a large-scale, open-source, and data-driven simulator for vehicle motion planning, including a comprehensive set of state-of-the-art planners (Scheel *et al.*, 2022; Renz *et al.*, 2022). Our analysis yields several surprising findings:

Open- and closed-loop evaluation are misaligned. Most learned planners are trained through the supervised learning task of forecasting the EV's future motion conditioned on a desired goal location. We refer to this setting as ego-forecasting (Codevilla *et al.*, 2019; Rhinehart *et al.*, 2019; Prakash *et al.*, 2021; Chitta *et al.*, 2021). In nuPlan, planners can be evaluated in two ways: (1) in open-loop evaluation, which measures ego-forecasting accuracy using distance-based metrics or (2) in closed-loop evaluation, which assesses the actual driving performance in simulation with metrics such as progress or collision rates. Open-loop evaluation lacks dynamic feedback and can have little correlation with closed-loop driving, as previously shown on the simplistic CARLA simulator (Codevilla *et al.*, 2018b; Dosovitskiy *et al.*, 2017). Our primary contribution lies in uncovering a *negative correlation* between both evaluation schemes. Learned planners excel at ego-forecasting but struggle to make safe closed-loop plans, whereas rule-based planners exhibit the opposite trend.

Rule-based planning generalizes. We surprisingly find that an established rule-based planning baseline from over twenty years ago (Treiber *et al.*, 2000) surpasses *all state-of-the-art learning-based methods* in terms of closed-loop evaluation metrics on our benchmark. This contradicts the prevalent motivating claim used in most research on learned planners that rule-based planning faces difficulties in generalization. This was previously only verified on simpler benchmarks (Zeng *et al.*, 2019; Scheel *et al.*, 2022; Renz *et al.*, 2022). As a result, most current work on learned planning only compares to other learned methods, ignoring rule-based baselines (Rhinehart *et al.*, 2019; Chitta *et al.*,

2022; Hu *et al.*, 2023c).

A centerline is all you need for ego-forecasting. We implement a naïve learned planning baseline which does not incorporate any input about other agents in the scene and merely extrapolates the ego state given a centerline representation of the desired route. This baseline *sets the new state-of-the-art* for open-loop evaluation on our benchmark. It does not require intricate scene representations (e.g. lane graphs, vectorized maps, rasterized maps, tokenized objects), which have been the central subject of inquiry in previous work (Scheel *et al.*, 2022; Renz *et al.*, 2022; Hallgarten *et al.*, 2023a). None of these prior studies considered a simple centerline-only representation as a baseline, perhaps due to its extraordinary simplicity.

Our contributions are as follows: (1) We demonstrate and analyze the misalignment between open- and closed-loop evaluation schemes in planning. (2) We propose a lightweight extension of `IDM` (Treiber *et al.*, 2000) with real-time capability that achieves state-of-the-art closed-loop performance. (3) We conduct experiments with an open-loop planner, which is only conditioned on the current dynamic state and a centerline, showing that it outperforms sophisticated models with complex input representations. (4) By combining both models into a hybrid planner, we establish a simple baseline that outperformed 24 other, often learning-based, competing approaches and claimed victory in the nuPlan challenge 2023.

6.2 Related Work

Rule-based planning. Rule-based planners offer a structured, interpretable decision-making framework (Treiber *et al.*, 2000; Thrun *et al.*, 2006; Bacha *et al.*, 2008; Leonard *et al.*, 2008; Urmson *et al.*, 2008; Chen *et al.*, 2015; Sauer *et al.*, 2018; Fan *et al.*, 2018; Sadat *et al.*, 2019). They employ explicit rules to determine an autonomous vehicle's behavior (e.g., brake when an object is straight ahead). A seminal approach in rule-based planning is the Intelligent Driver Model (`IDM` (Treiber *et al.*, 2000)), which is designed to follow a leading vehicle in traffic while maintaining a safe distance. There exist extensions of `IDM` (Kesting *et al.*, 2007) which focus on enabling lane changes on highways. However, this is not the goal of our work. Instead, we extend `IDM` by executing multiple policies with different hyperparameters, and scoring them to select the best option.

Prior work also combines rule-based decision-making with learned components, e.g., with learned agent forecasts (Chekroun *et al.*, 2023), affordance indicators (Chen *et al.*, 2015; Sauer *et al.*, 2018), cost-based imitation learning (Zeng *et al.*, 2019; Cui *et al.*, 2021; Casas *et al.*, 2021; Sadat *et al.*, 2020; Huang *et al.*, 2023a), or learning-based planning with rule-based safety filtering (Phan-Minh *et al.*, 2023). These hybrid planners often forecast future environmental states, enabling informed and contingent driving decisions. This forecasting can either be agent-centric (Karkus *et al.*, 2023; Danesh *et al.*, 2023; Huang *et al.*, 2022b), where trajectories are determined for each actor, or environment-centric (Zeng *et al.*, 2019; Sadat *et al.*, 2020; Casas *et al.*, 2021; Cui *et al.*, 2021; Wei *et al.*,

2021; Hu *et al.*, 2022), involving occupancy or cost maps. Additionally, forecasting can be conditioned on the ego-plan, modeling the EV's influence on the scene's future (Song *et al.*, 2020; Rhinehart *et al.*, 2021; Chen *et al.*, 2023e; Huang *et al.*, 2023b). We employ an agent-centric forecasting module that is considerably simpler than existing methods, allowing for its use as a starting point in the newly released nuPlan framework.

Ego-forecasting. Unlike predictive planning, ego-forecasting methods use observational data to directly determine the future trajectory. Ego-forecasting approaches include both end-to-end (E2E) methods (Chen *et al.*, 2023b) that utilize LiDAR scans (Rhinehart *et al.*, 2018a; Filos *et al.*, 2020), RGB images (Xu *et al.*, 2017; Chen *et al.*, 2020; Ohn-Bar *et al.*, 2020; Behl *et al.*, 2020; Chitta *et al.*, 2021; Wu *et al.*, 2022) or both (Prakash *et al.*, 2021; Chen and Krähenbühl, 2022; Chitta *et al.*, 2022; Jaeger *et al.*, 2023), as well as modular methods involving lower-dimensional inputs like Bird's Eye View (BEV) grids or state vectors (Sauer *et al.*, 2018; Hanselmann *et al.*, 2022; Renz *et al.*, 2022; Vitelli *et al.*, 2022; Pini *et al.*, 2022; Cheng *et al.*, 2023). A concurrent study introduces a naive MLP inputting the current dynamic state, yielding competitive ego-forecasting results on the nuScenes dataset (Caesar *et al.*, 2020) with no scene context input (Zhai *et al.*, 2023). Our findings complement these results, differing by evaluating long-term (8s) ego-forecasting in the challenging 2023 nuPlan challenge scenario test distribution (Caesar *et al.*, 2021). We show that in this setting, completely removing scene context (as in (Zhai *et al.*, 2023)) is harmful, whereas a simple centerline representation of the context is sufficient for strong open-loop performance.

6.3 Ego-forecasting and Planning are Misaligned

In this section, we provide the relevant background regarding the data-driven simulator nuPlan (Caesar *et al.*, 2021). We describe two baselines for a preliminary experiment to demonstrate that although ego-forecasting and planning are often considered related tasks, they are not well-aligned given their definitions on nuPlan. Improvements in one task can often lead to degradation in the other.

6.3.1 Background

nuPlan. The nuPlan simulator is the first publicly available real-world planning benchmark and enables rapid prototyping and testing of motion planners. nuPlan constructs a simulated environment as closely as possible to a real-world driving setting through data-driven simulation (Althoff *et al.*, 2017; Bergamini *et al.*, 2021; Suo *et al.*, 2021; Zhong *et al.*, 2023; Xu *et al.*, 2022a; Zhang *et al.*, 2023; Feng *et al.*, 2023). This method extracts road maps, traffic patterns, and object properties (positions, orientations, and speeds) from a pre-recorded dataset consisting of 1,300 hours of real-world driving. These elements are then used to initialize scenarios, which are 15-second simulations employed to assess open-loop and closed-loop driving performance. Hence, in simulation, our

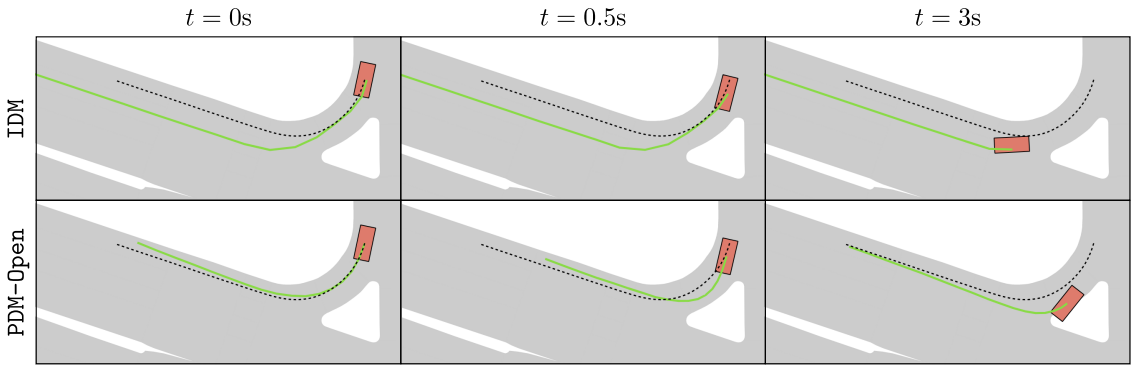


Figure 6.1: **Planning vs. ego-forecasting.** We present a nuPlan scenario, highlighting the driveable area in grey and the original human trajectory as a dashed black line. In each snapshot, we display the ego agent with its prediction. (Left) observe the significant displacement between the `IDM` prediction (constrained to a rule-based centerline) and the human trajectory, resulting in low open-loop scores. (Mid + right) after 0.5 seconds of simulation, the learned `PDM-Open` planner extrapolates its own errors and eventually veers off-road, leading to suboptimal closed-loop scores.

methods rely on access to detailed HD map information and ground-truth perception. i.e., no localization errors, map imperfections, or misdetections are considered. In open-loop simulation, the entire log is merely replayed (for both the EV and other actors). Conversely, in closed-loop simulation, the EV operates under the control of the planner being tested. There are two versions of closed-loop simulation: non-reactive, where all other actors are replayed along their original trajectory, and reactive, where other vehicles employ an `IDM` planner (Treiber *et al.*, 2000), which is detailed in the following.

Metrics. nuPlan offers three official evaluation metrics: open-loop score (OLS), closed-loop score non-reactive (CLS-NR), and closed-loop score reactive (CLS-R). Although CLS-NR and CLS-R are computed identically, they differ in background traffic behavior. Each score is a weighted average of sub-scores that are multiplied by a set of penalties. In OLS, the sub-scores account for displacement and heading errors, both average and final, over an extended period (8 seconds). Moreover, if the prediction error is above a threshold, the penalty results in an OLS score of zero for that scenario. Similarly, sub-scores in CLS comprise time-to-collision, progress along the experts' route, speed-limit compliance, and comfort. Multiplicative CLS penalties are at-fault collisions, driveable area or driving direction infringements and not making progress. These penalties result in substantial CLS reductions, mostly to a zero scenario score, e.g., when colliding with a vehicle. Notably, the CLS primarily relies on short-term actions rather than on consistent long-term planning. All scores (incl. OLS/CLS) range from 0 to 100, where higher scores are better.

Intelligent Driver Model. The simple planning baseline `IDM` (Treiber *et al.*, 2000) not

Parameter	Value	Description
v_0	v_{lane}	Desired velocity. Either the current speed-limit, or $v_{\text{lane}} = 10$ m/s if speed-limit not available.
s_0	1.0 m	Desired net distance to the leading agent.
T	1.5 s	Desired time headway to leading agent.
a	1.0 m/s^2	Maximum acceleration of ego vehicle
b	3.0 m/s^2	Maximum deceleration (positive) of ego vehicle
δ	4.0	Acceleration exponent.

Table 6.1: `IDM` Parameters.

only simulates the non-ego vehicles in the CLS-R evaluation of nuPlan, but also serves as a baseline for the ego-vehicle's planning. The nuPlan map is provided as a graph, with centerline segments functioning as nodes. After choosing a set of such nodes to follow via a graph search algorithm, `IDM` infers a longitudinal trajectory along the selected centerline. Given the current longitudinal position x, velocity v, and distance and relative velocity to the leading vehicle s and Δv along the centerline, it iteratively applies the following policy to calculate a longitudinal acceleration:

$$\frac{\mathrm{d}v}{\mathrm{d}t} = a\left(1 - \left(\frac{v}{v_0}\right)^{\delta} - \left(\frac{s^*}{s}\right)^2\right) \tag{6.1}$$

$$\text{with } s^* = s_0 + vT + \frac{v\Delta v}{2\sqrt{ab}}. \tag{6.2}$$

Intuitively, the policy uses an acceleration a unless the velocity is already close to v_0 or the leading vehicle is at a distance of only s^*. Our exact hyper-parameter choices can be found in Tab. 6.1.

6.3.2 Misalignment

Centerline-conditioned ego-forecasting. We now propose the Predictive Driver Model (Open), i.e., `PDM-Open`, which is a straightforward MLP designed to predict future waypoints. The inputs to this MLP are the centerline ($\mathbf{c}$) extracted by `IDM` and the ego history (X_{EV}). To accommodate the high speeds (reaching up to 15 m/s) and ego-forecasting horizons (extending to 8 seconds) observed in nuPlan, the centerline is sampled with a resolution of 1 meter up to a length of 120 meters. Meanwhile, the ego history $X_{EV} = \{s_t^{(EV)}\}_{t=t_0-t_h}^{t_0}$ incorporates states $s_t^{(EV)}$ including the positions, velocities, and accelerations of the vehicle over the previous two seconds ($t_h = 2$s), sampled at a rate of 5Hz. Both $\mathbf{c}$ and X_{EV} are linearly projected to feature vectors of size 512, concatenated, and input to the MLP $\phi_{\texttt{Open}}$ which has two 512-dimensional hidden layers. The output are the future waypoints for an 8-second horizon, spaced 0.5 seconds apart, expressed as

Method	Centerline	History	CLS-R ↑	CLS-NR ↑	OLS ↑
IDM (Treiber *et al.*, 2000) $a = 1.0$	✓	-	**77**	**76**	38
$a = 0.1$			54	66	48
PDM-Open	-	-	51	50	69
	-	✓	38	34	72
	✓	-	54	53	85
	✓	✓	54	50	**86**

Table 6.2: **OLS-CLS Tradeoff.** Baseline scores on nuPlan with different inputs.

$\mathbf{w}_{\texttt{Open}} = \phi_{\texttt{Open}}(\mathbf{c}, X_{\text{EV}})$. The model is trained using an L_1 loss on our training dataset of 177k samples (described in section 6.4). By design, PDM-Open is considerably simpler than existing learned planners (Scheel *et al.*, 2022; Hallgarten *et al.*, 2023a).

OLS vs. CLS. In Table 6.2, we benchmark the IDM and PDM-Open baselines using the nuPlan metrics. We present two IDM variants with different maximum acceleration values (the default $a = 1.0\text{ms}^{-2}$ and $a = 0.1\text{ms}^{-2}$) and four PDM-Open variants based on different inputs. We observe that reducing IDM's acceleration improves OLS but negatively impacts CLS. While IDM demonstrates strong closed-loop performance, PDM-Open outperforms IDM in open-loop even if it only uses the current ego state as input (first row). The past ego states (History) only yield little improvement and lead to a drop in CLS. Most importantly, adding the centerline significantly contributes to ego-forecasting performance. A clear trade-off between CLS and OLS indicates a misalignment between the goals of ego-forecasting and planning. This sort of inverse correlation on nuPlan is unanticipated, considering the increasing use of ego-forecasting in current planning literature (Rhinehart *et al.*, 2019; Scheel *et al.*, 2022; Hallgarten *et al.*, 2023a; Renz *et al.*, 2022). While ego-forecasting is not necessary for driving performance, the nuPlan challenge requires both a high OLS and CLS.

In figure 6.1, we illustrate the misalignment between the OLS and CLS metrics. In the depicted scenario, the rule-based IDM selects a different lane in comparison to the human driver. However, it maintains its position on the road throughout the simulation. This results in a high CLS yet a low OLS. Conversely, the learned PDM-Open generates predictions along the lane chosen by the human driver, thereby obtaining a high OLS. Nonetheless, as errors accumulate in its short-term predictions during the simulation (Ross *et al.*, 2011; Prakash *et al.*, 2020), the model's trajectory veers off the drivable area, culminating in a subpar CLS.

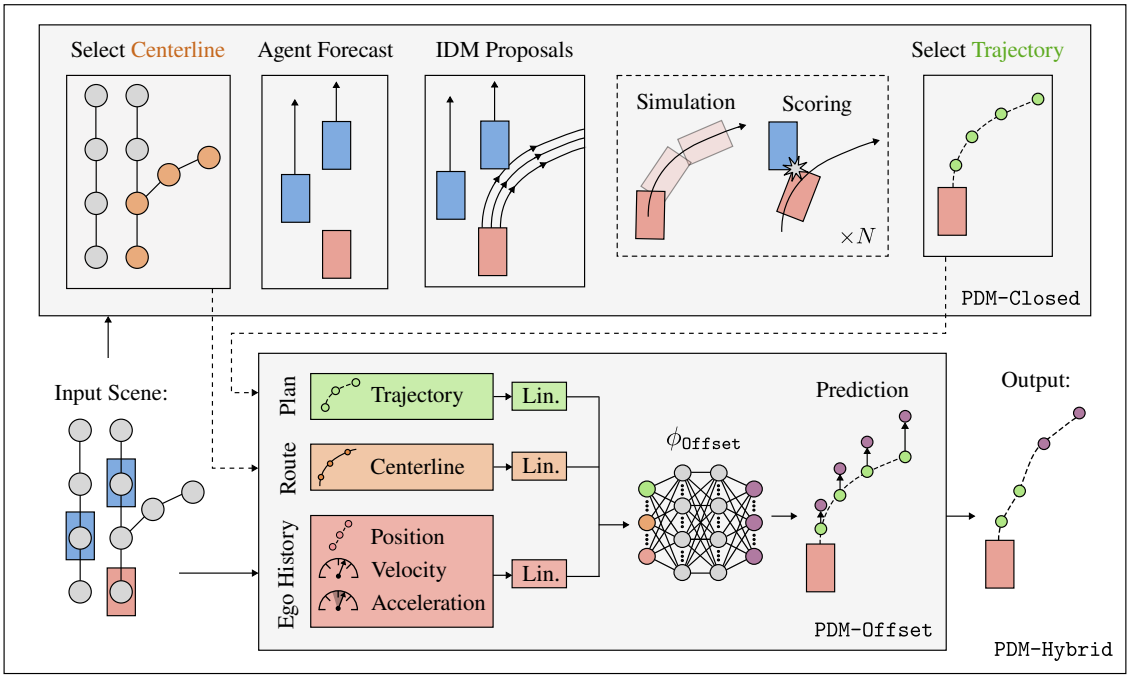


Figure 6.2: **Architecture.** `PDM-Closed` selects a centerline, forecasts the environment, and creates varying trajectory proposals, which are simulated and scored for trajectory selection. The `PDM-Hybrid` module predicts offsets using the `PDM-Closed` centerline, trajectory, and ego history, correcting only long-term waypoints and thereby limiting the learned model's influence in closed-loop simulation.

6.3.3 Methods

We now extend `IDM` by incorporating several concepts from model predictive control, including forecasting, proposals, simulation, scoring, and selection, as illustrated in figure 6.2 (top). We call this model `PDM-Closed`. Note that as a first step, we still require a graph search to find a sequence of lanes along the route and extract their centerline, as in the `IDM` planner.

Forecasting. In nuPlan, the simulator provides an orientation vector and speed for each dynamic agent such as a vehicle or pedestrian. We leverage a simple yet effective constant velocity forecasting (Poddar *et al.*, 2023; Schöller *et al.*, 2020; Wu *et al.*, 2023a) over the horizon F of 8 seconds at 10Hz.

Proposals. In the process of calibrating the `IDM` planner, we observed a trade-off when selecting a single value for the target speed hyperparameter (v_0), which either yielded aggressive driving behavior or insufficient progress across various scenarios. Consequently, we generate a set of trajectory proposals by implementing `IDM` policies at five distinct target speeds, namely, $\{20\%, 40\%, 60\%, 80\%, 100\%\}$ of the designated speed

limit. For each target speed, we also incorporate proposals with three lateral centerline offsets ($\pm$1m and 0m), thereby producing $N = 15$ proposals in total. To circumvent computational demands in subsequent stages, the proposals have a reduced horizon of H steps, which corresponds to 4 seconds at a 10Hz.

Simulation. Trajectories in nuPlan are simulated by iteratively retrieving actions from an LQR controller (Tassa *et al.*, 2014) and propagating the EV with a kinematic bicycle model (Rajamani, 2011; Polack *et al.*, 2017). We simulate the proposals with the same parameters and a faster re-implementation of this two-stage pipeline. Thereby, the proposals are evaluated based on the expected movement in closed-loop.

Scoring. Each simulated proposal is scored to favor traffic-rule compliance, progress, and comfort. By considering proposals with lateral and longitudinal variety, the planner can avoid collisions with agent forecasts and correct drift that may arise when the controller fails to accurately track the intended trajectory. Furthermore, our scoring function closely resembles the nuPlan evaluation metrics. However, we leverage a computationally efficient re-implementation of the metrics to meet the strict runtime requirements of the competition. The scoring considers at-fault collisions, drivable area infractions, and driving direction compliance as multiplicative metrics. Furthermore, the scoring evaluates progress, time-to-collision, and comfort as weighted metrics. We normalize the progress metric with the highest progress of a proposal free of multiplicative infractions. We use the same weights as nuPlan, but ignore speed-limit compliance and the binary no-progress metric since the `IDM` proposals are naturally bound to comply with the current speed limit, and the no-progress metric cannot be evaluated without privileged knowledge of the human expert's behavior.

Trajectory selection. Finally, `PDM-Closed` selects the highest-scoring proposal which is extended to the complete forecasting horizon F with the corresponding `IDM` policy. If the best trajectory is expected to collide within 2 seconds, the output is overwritten with an emergency brake maneuver.

Enhancing long-horizon accuracy. To integrate the accurate ego-forecasting capabilities of `PDM-Open` with the precise short-term actions of `PDM-Closed`, we now propose a hybrid version of `PDM`, i.e., `PDM-Hybrid`. Specifically, `PDM-Hybrid` uses a learned module `PDM-Offset` to predict offsets to waypoints from `PDM-Closed`, as shown in figure 6.2 (bottom).

In practice, the LQR controller used in nuPlan relies exclusively on the first 2 seconds of the trajectory when determining actions in closed-loop. Therefore, applying the correction only to long-term waypoints (i.e., beyond 2 seconds by default, which we refer to as the correction horizon C) allows `PDM-Hybrid` to maintain closed-loop planning performance. The final planner outputs waypoints (up to the forecasting horizon F) $\{\mathbf{w}^t_{\texttt{Hybrid}}\}^F_{t=0}$ that are given by:

$$\mathbf{w}^t_{\texttt{Hybrid}} = \mathbf{w}^t_{\texttt{Closed}} + \mathbb{1}_{[t>C]}\phi^t_{\texttt{Offset}}(\mathbf{w}_{\texttt{Closed}}, \mathbf{c}, X_{\text{EV}}). \tag{6.3}$$

Where $\mathbf{c}$ and X_{EV} are the centerline and history (identical to the inputs of `PDM-Open`).

Method	Rep.	CLS-R ↑	CLS-NR ↑	OLS ↑	Time ↓
Urban Driver (Scheel *et al.*, 2022)	Polygon	50	53	82	64
GC-PGP (Hallgarten *et al.*, 2023a)	Graph	55	59	83	100
PlanCNN (Renz *et al.*, 2022)	Raster	72	73	64	43
IDM (Treiber *et al.*, 2000)	Centerline	77	76	38	27
PDM-Open	Centerline	54	50	**86**	**7**
PDM-Closed	Centerline	**92**	**93**	42	91
PDM-Hybrid	Centerline	**92**	**93**	84	96
PDM-Hybrid*	Graph	**92**	**93**	84	172
Log Replay	*GT*	*80*	*94*	*100*	-

Table 6.3: **Val14 benchmark.** We show the closed-loop score reactive/non-reactive (CLS-R/CLS-NR), open loop score (OLS) and runtime in ms for several planners. We specify the input representation (Rep.) used by each planner. PDM-Hybrid accomplishes strong ego-forecasting (OLS) and planning (CLS). *This is a preliminary version of PDM-Hybrid that combined PDM-Closed with GC-PGP (Hallgarten *et al.*, 2023a), and was used in our online leaderboard submission (Table 6.4).

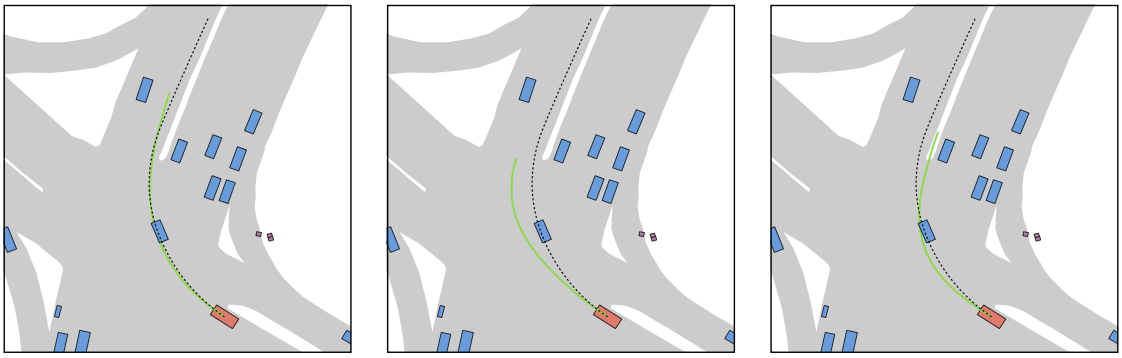
$\{\mathbf{w}^t_{\texttt{Closed}}\}_{t=0}^{F}$ are the PDM-Closed waypoints added to the hybrid approach, and $\phi_{\texttt{Offset}}$ is an MLP. Its architecture is identical to $\phi_{\texttt{Open}}$ except for an extra linear projection to accommodate $\mathbf{w}_{\texttt{Closed}}$ as an additional input.

6.4 Experiments

We now outline our proposed benchmark and highlight the driving performance of our approach.

Val14 benchmark. We offer standardized data splits for training and evaluation. Training uses all 70 scenario types from nuPlan, restricted to a maximum of 4k scenarios per type, resulting in ∼177k training scenarios. For evaluation, we use 100 scenarios of the 14 scenario types considered by the leaderboard, totaling 1,118 scenarios. Despite minor imbalance (all 14 types do not have 100 available scenarios), our validation split aligns with the online leaderboard evaluation (Table 6.3 and Table 6.4), confirming the suitability of our Val14 benchmark as a proxy for the online test set.

Baselines. We include several additional state-of-the-art approaches adopting ego-forecasting for planning in our study. Urban Driver (Scheel *et al.*, 2022) encodes polygons with PointNet layers and predicts trajectories with a linear layer after a multi-head attention block. Our study uses an implementation of Urban Driver trained in the open-loop setting. GC-PGP (Hallgarten *et al.*, 2023a) clusters trajectory proposals based on route-constrained lane-graph traversals before returning the most likely cluster center. PlanCNN (Renz *et al.*, 2022) predicts waypoints using a CNN from rasterized grid features without an ego state input. It shares several similarities to ChauffeurNet (Bansal *et al.*,



(a) PDM-Open (b) GC-PGP (c) Urban Driver

Figure 6.3: nuPlan scenario (6bc4abef0de65d50), with the drivable area (light-gray), the ego-vehicle (red), the planner prediction (green), and human trajectory (dashed-black). The (a) PDM-Open model is accurate for ego-forecasting that requires lane-following over a long horizon. In comparison, (b) GC-PGP (Hallgarten *et al.*, 2023a) or (c) Urban Driver (Scheel *et al.*, 2022) have significantly higher displacements, leading to a zero OLS.

2018), a seminal work in the field. A preliminary version of PDM-Hybrid, which won the nuPlan competition, used GC-PGP(see Ch. 5) as its ego-forecasting component, and we include this as a baseline.

Results. Our results are presented in Table 6.3. PlanCNN achieves the best CLS among learned planners, possibly due to its design choice of removing ego state from input, trading OLS for enhanced CLS. Contrary to the community's growing preference for graph- and vector-based scene representations in prediction and planning (Deo *et al.*, 2022; Renz *et al.*, 2022; Nayakanti *et al.*, 2022; Cui *et al.*, 2023), these results show no clear disadvantage of raster representations for the closed-loop task, with PlanCNN also offering a lower runtime. Surprisingly, the simplest rule-based approach in our study, IDM, outperforms the best learned planner, PlanCNN. Moreover, we observe PDM-Closed's advantages over IDM in terms of CLS: an improvement from 76-77 to 92-93 as a result of the ideas from section 6.3. Surprisingly, PDM-Open achieves the highest OLS of 86 with a runtime of only 7ms using only a centerline and the ego state as input. We observe that PDM-Open improves on other methods in accurate long-horizon lane-following, as detailed further in Fig. 6.3. Next, despite PDM-Closed's unsatisfactory 42 OLS, PDM-Hybrid successfully combines PDM-Closed with PDM-Open. Both the centerline and graph versions of PDM-Hybrid achieve identical scores in our evaluation. However, the final centerline version, using PDM-Open instead of GC-PGP, is more efficient during inference. Finally, the privileged approach of outputting the ground-truth ego future trajectory (log replay) fails to achieve a perfect CLS, in part due to the nuPlan framework's LQR controller occasionally drifting from the provided trajectory. PDM-Hybrid compensates for this by evaluating proposals based on the expected controller outcome, causing it to

Method	CLS-R ↑	CLS-NR ↑	OLS ↑	Score ↑
PDM-Hybrid*	**93**	**93**	83	**90**
hoplan	89	88	85	87
pegasus_multi_path	82	85	**88**	85
Urban Driver (Scheel *et al.*, 2022)	68	70	86	75
IDM (Treiber *et al.*, 2000)	72	75	29	59

Table 6.4: **2023 nuPlan Challenge.**

match/outperform log replay in closed-loop evaluation.

Challenge. The 2023 nuPlan challenge saw the preliminary (graph) version of PDM-Hybrid rank first out of 25 participating teams. The leaderboard considers the mean of CLS-R, CLS-NR, and OLS. While open-loop performance lagged slightly, closed-loop performance excelled, resulting in an overall state-of-the-art score. Unfortunately, due to the closure of the leaderboard, our final (centerline) version of PDM-Hybrid that replaces GC-PGP with the simpler PDM-Open module could not be benchmarked. All top contenders combined learned ego-forecasting with rule-based post-solvers or post-processing to boost CLS performance for the challenge (Hu *et al.*, 2023b; Xi *et al.*, 2023b; Huang *et al.*, 2023c). Thus, we expect to see more hybrid approaches in the future.

Importantly, near identical scores were recorded for our submission on both our Val14 benchmark (Table 6.3) and the official leaderboard (Table 6.4). Note that the Urban Driver and IDM results on the leaderboard are provided by the nuPlan team, so they likely use different training data and hyper-parameters than our implementations from Table 6.3.

Ablation Study. We delve into our design choices through an ablation study in Table 6.5. Table 6.5a displays PDM-Hybrid's closed-loop score reactive (CLS-R) and open-loop score (OLS) with varied correction horizons (C) from 0s to 3s. Applying the waypoint correction to all waypoints (i.e., $C = 0$), outperforms PDM-Open in OLS (87 vs. 86, see Table 6.3) but leads to a substantial drop in CLS-R compared to the default value of $C = 2$. On the other hand, a noticeable OLS decline occurs when initiating corrections deeper into the trajectory (e.g., $C = 3$), with minimal impact on CLS-R.

For PDM-Closed (Table 6.5b), we compare CLS-R and runtime (ms) with the base planner across three scenarios: removing lateral centerline offsets ("lat."), longitudinal IDM proposals ("lon."), and environment forecasting ("cast."). Our analysis reveals that eliminating proposals diminishes CLS-R effectiveness but accelerates runtimes. Performance significantly drops when excluding the forecasting used for creating and evaluating proposals. However, the runtime remains nearly identical, showing the effectiveness of the simple forecasting mechanism.

As for PDM-Open (Table 6.5c), we test three variations: a shorter centerline (30m vs. 120m), a coarser centerline (every 10m vs. 1m), and a smaller MLP with a reduced

(a) PDM-Hybrid

C	CLS-R ↑	OLS ↑
0.0s	58	**87**
2.0s	**92**	84
2.5s	**92**	84
3.0s	**92**	72

(b) PDM-Closed

Method	CLS-R ↑	Time ↓
Base	**92**	91
No lat.	89	**55**
No lon.	88	64
No cast.	86	90

(c) PDM-Open

Method	OLS ↑
Baseline	**86**
Shorter centerline	84
Coarser centerline	**86**
Smaller MLP	84

Table 6.5: **Ablation Study.** We show the closed-loop score reactive (CLS-R), open loop score (OLS) and runtime in ms. We investigate (a) varying correction horizons for PDM-Hybrid, (b) ignoring sub-modules of PDM-Closed, and (c) the effects of input and architecture choices on PDM-Open. The default configuration, highlighted in gray, achieves the best trade-offs.

hidden dimension (from 512 to 256). Both a smaller MLP and a reduced centerline length lead to performance degradation, but the impact remains relatively minor compared to disregarding the centerline altogether (Table 6.2, OLS=72). Meanwhile, the impact of a coarser centerline is negligible.

6.5 Discussion

Although rule-based planning is often criticized for its limited generalization, our results demonstrate strong performance in the closed-loop nuPlan task which best resembles real-world evaluation. Notably, open-loop success in part requires a *trade-off in closed-loop performance*. Consequently, imitation-trained ego-forecasting methods fare poorly in closed-loop. This suggests that rule-based planners remain promising and warrant further exploration. At the same time, given their poor performance out-of-the-box, there is room for improvement in imitation-based methods on nuPlan.

Integrating the strengths of closed-loop planning and open-loop ego-forecasting, we present a hybrid model. However, this does not enhance closed-loop driving performance; instead, it boosts open-loop performance *while executing identical driving maneuvers*. We conclude that considering precise open-loop ego-forecasting as a prerequisite for achieving long-term planning goals is misleading.

Acknowledging the potential importance of ego-forecasting for interpretability and assessing human-like behavior, we propose focusing this evaluation on the short horizon (e.g., 2 seconds) relevant for closed-loop driving. The current nuPlan OLS definition, requiring a unimodal 8-second ego-forecast, may only be useful for *alternate applications*, like setting goals for background agents in data-driven traffic simulations or allocating computational resources better, e.g. to prioritize perception or prediction in areas the ego-vehicle is expected to traverse. We discourage the use of open-loop metrics as a primary indicator of planning performance (Hu *et al.*, 2023c).

Limitations. While we significantly improve upon the established IDM model, PDM still

does not execute lane-change maneuvers. Lane change attempts often lead to collisions when the ego-vehicle is between two lanes, resulting in a high penalty as per the nuPlan metrics. PDM relies on HD maps and precise offboard perception (Qi *et al.*, 2021; Caesar *et al.*, 2021) that may be unavailable in real-world driving situations. While real-world deployment was demonstrated for learning-based methods (Scheel *et al.*, 2022; Phan-Minh *et al.*, 2023; Bansal *et al.*, 2018), it remains a significant challenge for rule-based approaches. Moreover, our experiments, aside from the held-out test set, have not specifically evaluated the model's generalization capabilities when encountering distributional shifts, such as unseen towns or novel scenario types. They were all conducted on a single simulator, nuPlan. Therefore, it is important to recognize the limitations inherent in nuPlan's data-driven simulation approach. When a planner advances more rapidly than the human driving log, objects materialize abruptly in front of the ego-vehicle during simulation. For CLS-NR, vehicles move independently as observed in reality, disregarding the ego agent, leading to excessively aggressive behavior. Conversely, CLS-R background agents rely on IDM and adhere strictly to the centerline, leading to unrealistically passive behavior. We see high value in developing a more refined reactive environment for future work.

Conclusion. In this chapter, we identify prevalent misconceptions in learning-based vehicle motion planning. Based on our insights, we introduce PDM-Hybrid, which builds upon IDM and combines it with a learned ego-forecasting component. It surpassed a comprehensive set of competitors and claimed victory in the 2023 nuPlan competition. Notably, it almost matches the human expert's performance in the nonreactive closed-loop simulation and achieves excellent results in the reactive simulation, successfully navigating most of the tested scenarios.

Chapter 7

Generalization to Realistic and Interactive Long-Tail Scenarios

7.1 Introduction

Our results presented in the previous chapter show that simple rule-based planners outperform learning-based methods in the recently proposed closed-loop nuPlan benchmark (Caesar *et al.*, 2021). Even more surprisingly, the rule-based method PDM-Closed achieves a nearly perfect score, suggesting that it is capable of tackling the enormous challenge of real-world driving. In this chapter, we challenge this conception. The fact that PDM-Closed (Dauner *et al.*, 2023) is a reactive centerline planner, unable to do lane changes or interact in a targeted way with surrounding traffic agents, underlines that the nuPlan benchmark mostly covers basic driving scenarios. While solving these is a fundamental requirement for motion planning methods, it falls short of proving that they generalize to rare scenarios from the long-tailed distribution of potential real-world scenarios. However, generalization to previously unseen driving scenarios is crucial to achieving autonomy. Therefore, we propose a novel benchmark to evaluate vehicle motion planning methods in highly interactive and rare scenarios.

The interPlan benchmark. Interactive traffic maneuvers are characterized by mutual influence between traffic participants and the ego-vehicle. Understanding and leveraging these bi-directional interactions is crucial for smooth progress. Thus, we take nuPlan scenarios as a starting point for our benchmark and augment them with additional agents, obstacles, or alternative navigation goals, creating realistic long-tail scenarios. In total, our benchmark comprises 80 scenarios covering the following situations: Nudging around parked vehicles, overtaking obstacles, passing a construction zone, passing an accident site, facing jaywalkers, as well as lane changes in low, medium, and high traffic density.

State-of-the-art methods do not generalize to long-tail scenarios. We conduct experiments with an exhaustive set of state-of-the-art planning methods and reveal critical shortcomings in their ability to generalize to difficult unseen scenarios. Rule-based vehicle motion planning methods rely on predefined rules and behaviors to navigate the vehicle. They fail to handle complex and uncommon driving scenarios not covered by the limited rules. On the other hand, learning-based methods are impaired by a lack of robustness

	nuPlan (Caesar *et al.*, 2021)	Carla (Dosovitskiy *et al.*, 2017)	interPlan **(Ours)**
Task	Planning	E2E Driving	Planning
based on real-world scenarios	✓	✗	✓
diverse traffic agents	✗	✗	✓
long-tail scenarios	✗	✓	✓
interactive lane-changes	✗	✓	✓
scenario-generation interface	✗	✓	✓
LLM Interface	✗	✗	✓

Table 7.1: **Comparative summary of planning benchmarks**

and thus cannot deliver the promise of better generalization.

LLM-based planning. Foundation models, especially Large Language Models (LLMs), have shown outstanding world-understanding and generalization ability. This makes them tailored to analyze complex driving environments and make informed and well-founded decisions. Therefore, we propose LLM-based planning baselines for our novel benchmark and analyze their capabilities in closed-loop simulation.

Contributions. We summarize our contributions as follows:

- We propose the *interPlan* benchmark, a publicly available closed-loop driving benchmark focused on highly interactive and difficult scenarios. We evaluate and analyze a comprehensive set of state-of-the-art methods.
- We demonstrate that even though some methods achieve excellent results in common driving scenarios, they fail in complex long-tail situations, e.g., passing accident sites.
- Fueled by the rising interest in motion planning based on LLMs, we implement GPT-Driver (Mao *et al.*, 2023a) as a baseline and challenge its abilities.
- We propose a novel two-stage planner that combines an LLM-based behavior planner with a downstream rule-based motion planner. It led to a new state-of-the-art on the novel benchmark and serves as a strong baseline for future research.
- We release code for both the *interPlan* benchmark as well as integration of LLMs with the nuPlan framework, hoping to facilitate further research on the topic.

7.2 Related Work

Closed-loop vehicle motion planning. Training a learning-based planner through imitation learning is a straightforward and widely adopted approach (Chitta *et al.*, 2021,

2022; Zeng *et al.*, 2019; Rhinehart *et al.*, 2019; Codevilla *et al.*, 2019, 2018a) pioneered by ALVINN (Pomerleau, 1988). Rule-based methods, on the other hand, employ a hand-designed decision-making framework with interpretable rules (Thrun *et al.*, 2006; Bacha *et al.*, 2008; Leonard *et al.*, 2008; Urmson *et al.*, 2008; Chen *et al.*, 2015; Sauer *et al.*, 2018; Fan *et al.*, 2018). For instance, the Intelligent Driver Model (IDM) (Treiber *et al.*, 2000) always follows a lane-centerline while maintaining a safe distance to the leading vehicle. Two predominant methods exist in evaluating vehicle motion planners. Open-loop ego forecasting compares the planned trajectory to an expert's ground-truth trajectory using distance-based metrics. In contrast, closed-loop evaluation involves simulating the planned trajectory with realistic environmental feedback, such as vehicle dynamics and reactions of surrounding traffic agents. To assess the interactive behavior in complex and interactive scenarios, closed-loop evaluation is indispensable, particularly as open-loop ego-forecasting evaluation was shown to be misaligned with closed-loop driving (Dauner *et al.*, 2023; Codevilla *et al.*, 2018b). The most widely adopted driving simulators are CARLA (Dosovitskiy *et al.*, 2017) and nuPlan (Caesar *et al.*, 2021). The simplistic CARLA (Dosovitskiy *et al.*, 2017) simulator uses synthetic data, which is not guaranteed to produce realistic driving situations. On the other hand, nuPlan (Caesar *et al.*, 2021) is based on real-world data, which comes at the cost of undersampling rare and critical scenarios, since collecting them is prohibitive due to the cost and danger involved. For instance, obtaining recordings of an emergency break caused by a child running on the street is dangerous and unethical. While the Carla Leaderboard fueled research for end-to-end (E2E) methods that directly process sensor information (Chen and Krähenbühl, 2022; Chitta *et al.*, 2022; Zeng *et al.*, 2019; Jaeger *et al.*, 2023; Shao *et al.*, 2023b; Wu *et al.*, 2022; Shao *et al.*, 2023a), the nuPlan competition focused on modular planners that use outputs of upstream perception modules (e.g., bounding boxes, maps) as inputs (Scheel *et al.*, 2022; Renz *et al.*, 2022; Hallgarten *et al.*, 2023a; Huang *et al.*, 2023b,a; Cheng *et al.*, 2023). In this work, we make use of the nuPlan closed-loop simulation to assess planners independently from perception noise. In order to assess generalization capabilities to rare driving situations, our proposed *interPlan* benchmark is made up of augmented real-world scenarios.

Benchmarking in complex interactive scenarios. Autonomous driving systems need to meet high safety standards. In particular, they must be extraordinarily robust when confronted with unseen scenarios. Various approaches exist to generate safety-critical scenarios for the synthetic Carla simulator (Dosovitskiy *et al.*, 2017; Ding *et al.*, 2020, 2021b; Wang *et al.*, 2021). These methods focus on robustness in the face of rule-breaking or assertive agents and behaviors that mislead the EV into questionable decisions. Other approaches include gradient-based optimization (Hanselmann *et al.*, 2022; Rempe *et al.*, 2022) and prior-knowledge based scenario construction (Bagschik *et al.*, 2018; McDuff *et al.*, 2022; Menzel *et al.*, 2018). Similarly, we augment driving scenarios to assess the interactive behavior of planning methods, facing a variety of traffic conditions ranging from sparse conservative traffic to high-density assertive traffic. Moreover, we also test scenarios that are not safety-critical but require profound reasoning and robust

generalization, such as passing an accident site or a construction zone. In contrast to previous work, we base our scenario generation on real-world scenarios from the widely adopted nuPlan dataset and simulator. We thus benefit from an unprecedented amount of typical driving scenarios for training before testing a planner's generalization to long-tail scenarios.

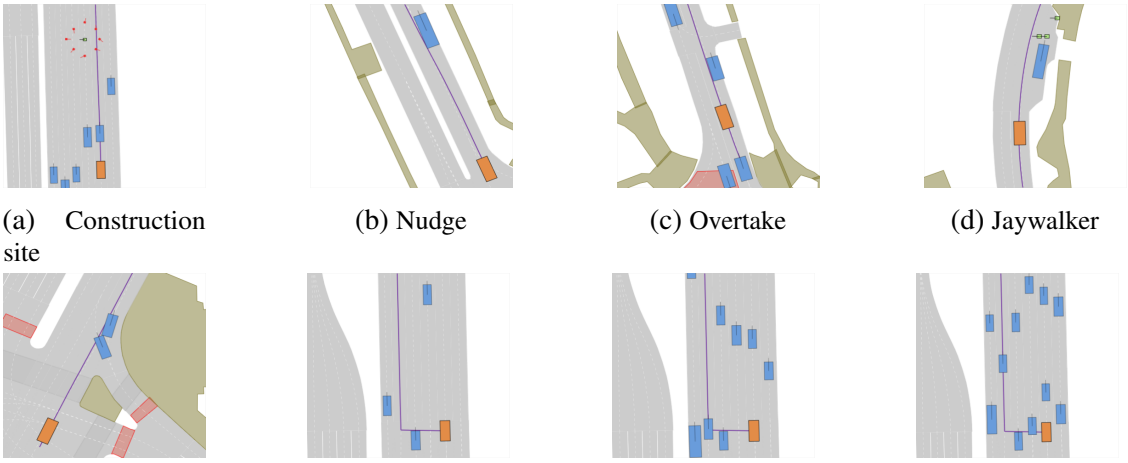
LLM-based vehicle motion planning. With the success of LLMs, a vital field of research emerged from applying them to vehicle motion planning. GPT-Driver (Mao *et al.*, 2023a) proposes to model motion planning as a language modeling task. By feeding outputs of a perception module (Hu *et al.*, 2023c) into a prompt and instructing the LLM to perform chain-of-thought reasoning to ultimately output a safe motion trajectory, it achieves state-of-the-art performance on the open-loop nuScenes benchmark (Caesar *et al.*, 2020). This was extended by AgentDriver (Mao *et al.*, 2023b), which uses another LLM to generate the task prompt. We introduce a hybrid approach that combines the outstanding world understanding and common sense of an LLM-based behavior planner with the excellent robustness of a rule-based motion planner. While prior work demonstrated strong reasoning capabilities of LLMs in traffic scenarios (Sha *et al.*, 2023; Chen *et al.*, 2023a; Kim *et al.*, 2020; Jin *et al.*, 2023), methods are often only evaluated in open-loop evaluation (Sun *et al.*, 2023; Mao *et al.*, 2023a; Hu *et al.*, 2023c; Mao *et al.*, 2023b; Sima *et al.*, 2023), in the synthetic Carla simulator (Dosovitskiy *et al.*, 2017; Wang *et al.*, 2023a), or the simplistic HighwayEnv environment (Leurent, 2018; Fu *et al.*, 2024; Wang *et al.*, 2023c). In contrast, our closed-loop *interPlan* benchmark is based on augmented real-world data, covers 80 complex driving scenarios, and includes a comprehensive set of baselines.

7.3 Realistic Scenario Generation

7.3.1 Problem Formulation

Generalizing to difficult unseen scenarios is a crucial capability to enable real-world autonomy. However, extracting useful scenarios for benchmarking from real-world recordings is practically infeasible due to their low probability. Staging them explicitly in real contexts is costly and often morally questionable. At the same time, generating such scenarios from scratch raises the question of realism. Therefore, we re-use scenarios from a large-scale real-world dataset and augment them to create difficult and rare scenarios that represent the long tails of the distribution of real-world situations, such as encountering construction zones, jaywalking pedestrians, and accident sites. We chose the large-scale nuPlan dataset (Caesar *et al.*, 2021) and the respective closed-loop simulator with reactive agents.

Being able to make plans in anticipation of surrounding agents' intentions and reactions is a fundamental pillar of real-world driving and essential to solving scenarios like unprotected turns, overtakes, or lane changes (Rhinehart *et al.*, 2021). Therefore, we also



(a) Construction site (b) Nudge (c) Overtake (d) Jaywalker

(e) Accident site (f) Lane-change (LTD) (g) Lane-change (MTD) (h) Lane-change (HTD)

Figure 7.1: **interPlan scenario types.** The ego vehicle and its navigation route are shown in orange and purple, surrounding vehicles and pedestrians are blue and green respectively. Traffic cones and stop lines are depicted in red.

evaluate state-of-the-art planning methods in highly interactive scenarios, such as highway merges at different traffic densities or overtaking parked vehicles. In the following, we describe how we augment the nuPlan scenarios to generate challenging interactive and rare scenarios to build our *interPlan* benchmark.

7.3.2 Scenario Generation

Realistic long-tail scenarios. In our benchmark, we aim to test long-tail scenarios, which are underrepresented in real-world driving and thus might not occur in large-scale training data. Such scenarios include encountering a lane blocked by construction zones (cf. Fig. 7.1a), nudging around (cf. Fig. 7.1b) or overtaking (cf. Fig. 7.1c) a parked vehicle, and facing jaywalkers at bus stops (cf. Fig. 7.1d). We generate these scenarios for our benchmark by adding objects such as traffic cones, parked vehicles, or stopped busses to the original nuPlan scenarios. Moreover, pedestrians who start crossing the street right when the EV approaches test the planner's ability to anticipate such non-compliant behaviors. In addition, we include scenarios where the planner has to navigate around an accident site (cf. Fig. 7.1e). This is particularly challenging because stationary touching vehicles on intersections or lanes are rarely, if at all, found in real-world datasets. Thus, sophisticated reasoning is required to understand that these vehicles have collided and cannot be expected to start moving, so they must be overtaken carefully. We design the crash sites in such a way that they represent common accidents such as rear-end collisions or collisions with crossing traffic.

Interactive lane-change scenarios. A major shortcoming of the original nuPlan benchmark is that it can be solved almost perfectly by simple lane-following, thus with little interaction with surrounding traffic. For instance, the challenge-winning method PDM-Closed (Dauner *et al.*, 2023) is by design unable to perform lane changes but achieves a nearly perfect score. To counteract this, we incentivize lane changes by augmenting the goal of original nuPlan scenarios. For instance, in the scenario shown in Fig. 7.1f, the EV has to do 3 lane changes. Moreover, we initialize the scenarios with different levels of traffic densities, i.e., low (LTD), medium (MTD), and high traffic density (HTD), by randomly spawning traffic agents around the EV.

Augmented traffic agent behavior. Real-world drivers exhibit diverse behaviors ranging from conservative and careful driving to assertive and reckless maneuvers. Planning methods have to be able to handle these varying conditions, especially without knowing what they are about to encounter. Our benchmark tests the ability of planning methods to make safe decisions under these varying conditions by applying different policies to control the surrounding traffic agents. Hence, in addition to nuPlan's default reactive IDM policy, which makes SVs break as soon as the EV enters their lane, we implement an assertive policy, which only reacts to the EV when it has fully merged into the lane. Simulation of merging scenarios can be carried out with all agents being controlled by the conservative reactive policy, the assertive policy, or a mixed policy, which randomly assigns a policy to each agent. The lane-change scenarios in our benchmark combine these traffic agent policies with different traffic densities. Thus, lane changes in HTD require complex interaction and potentially even prioritizing safety over progress. At the same time, LTD scenarios test a planner's general ability to merge into another lane.

7.3.3 The interPlan Benchmark

We test an exhaustive list of state-of-the-art planners, including learning-based (PDM-Open (Dauner *et al.*, 2023), UrbanDriver (Scheel *et al.*, 2022), GC-PGP (Hallgarten *et al.*, 2023a), Gameformer (Huang *et al.*, 2023b), DTPP (Huang *et al.*, 2023a)) and rule-based planners (IDM (Treiber *et al.*, 2000), IDM+MOBIL (Kesting *et al.*, 2007), PDM-Closed (Dauner *et al.*, 2023)). While some strictly follow lane-centerlines, others are not limited regarding their planning space. We intend to evaluate how well these planners can generalize to unseen scenarios, which is an indispensable capability for safe deployment. As a consequence, we intentionally test all planners in a zero-shot setting, i.e., they are trained on the large-scale real-world nuPlan train set without fine-tuning on the augmented benchmark scenarios. For the *interPlan* benchmark, we select 10 scenarios for each of the following types: Avoiding a construction zone (Constr.), encountering an accident (Acc.), jaywalking pedestrian (Jayw.), nudging around a parked vehicle (Nudge), overtaking a parked vehicle through the oncoming lane (Overt.), and lane changes in low (LTD), medium (MTD) and high (HTD) traffic density. Among the 10 LTD, MTD, and HTD scenarios, there are 3, 3, and 4 with conservative agents, assertive agents, and mixed agents, respectively.

Scenario type	Val14 mining	interPlan (Ours)
Total scenarios	51	80
Noisy objects	7	0
Unsafe initialization	13	0
Unsafe IDM behavior	8	0
Basic driving scenarios	13	0
Jaywalkers (non-react./react.)	6 / 0	0 / 10
Non-reactive merge	4	0
Interactive lane change	0	30
Nudge	0	10
Overtake	0	10
Construction site	0	10
Accident site	0	10

Table 7.2: **Comparison of interPlan to Val14 mining.**

7.3.4 Metrics

The *interPlan* benchmark builds upon the established nuPlan metrics, which comprise compliance with driving direction and drivable area, speed limit, as well as safety (collisions, time-to-collision), and comfort (accelerations, jerk). Metrics for comfort, speeding, progress, and keeping the TTC within bounds are aggregated into a weighted average. The other metrics (collisions, compliance with driving direction and area, being stationary for too long) do not contribute to the weighted average, but immediately reduce the score to 0 if performance is below a threshold. E.g., if the vehicle causes a collision, the entire scenario will be evaluated with a score of 0. We extend this driving score as follows: We measure which percentage of lane changes required to get to the goal was completed and include it in the weighted average before applying the penalties. Moreover, we add an additional multiplicative penalty based on the minimal progress needed to pass the obstacles (parked cars, construction zones, etc.). Hence, if the vehicle gets stuck in front of an obstacle, the scenario is evaluated with a score of 0%. Finally, we deactivate the driving direction compliance penalty for overtakes (Overt.) and accident sites (Acc.), where the vehicle has to go in an oncoming lane to pass the obstacle.

7.3.5 Comparison to Val14 Mining

A straightforward alternative to our scenario construction method is mining a large dataset for challenging scenarios (Cheng *et al.*, 2023). We mine the Val14 test split with the state-of-the-art PDM-Closed planner and identify 51 scenarios where the score is below 60%. See Tab. 7.2 for a comparison between the mined scenarios and *interPlan*.
20 of these scenarios turn out to be simulation failures, with initialization of the EV too close to an obstacle or even off-road (13) or where the sudden appearance of an object leads to an imminent collision (7). Another eight scenarios contain unrealistically unsafe

	Method	Val14	interPlan	Constr.	Acc.	Jayw.	Nudge	Overt.	LTD	MTD	HTD	Driv.	Goal	No-Col.
learned	Urban Driver	50	4	0	0	0	0	0	0	29	0	30	30	47
	GC-PGP	55	10	0	0	0	0	0	18	16	44	73	17	67
	GameFormer	75	11	0	0	48	0	0	0	20	21	30	30	87
	PDM-Open	54	25	13	0	56	36	8	29	29	26	60	40	43
	DTPP	73	25	18	**18**	44	10	0	40	36	34	60	33	93
rule	IDM	77	31	0	0	**66**	0	0	61	61	61	**100**	0	**100**
	IDM+MOBIL	75	31	**21**	0	**66**	0	0	**71**	21	**70**	93	**52**	80
	PDM-Closed	**92**	**42**	18	0	48	74	**9**	62	**62**	62	**100**	0	**100**

Table 7.3: **The interPlan benchmark.** Val14 denotes the score reported on common scenarios for reference. Besides the aggregated interPlan score, we also report the scores on each scenario type. The sub-scores drivable area compliance (Driv.), reaching the goal lane (Goal), and collision avoidance (No-Col.) refer only to the lane-change scenarios (LTD, MTD, HTD).

behavior of the IDM agents, such as running a red light.
In 13 of the remaining 23 scenarios, the low performance is merely caused by basic planner failures, such as getting stuck at a stop line or slightly going offroad at a tight turn. Thus, the scenarios are not challenging or of particular interest. Finally, collisions with pedestrians appear in six scenarios, and collisions during merges in four. In nuPlan simulations, pedestrians always non-reactively follow their recorded logs, which can lead to curious collisions if the controlled vehicle does not exactly follow the recording vehicle's path. Conversely, in our jaywalker scenarios, the movement of pedestrians is triggered by the approaching EV to always allow a reaction, and the locations are selected in such a way that jaywalkers can be anticipated (e.g., at bus stops). Finally, the four collisions at merges are related to a weakness of the IDM policy used to simulate surrounding agents: They do not react to vehicles in the neighboring lane, not even if the lanes are about to merge. Our lane-change scenarios force the EV to enter the lane of a neighboring IDM agent. By entering the neighboring lane, the EV becomes visible to the IDM so that it is able to react to it. Because different agent policies are employed (cf.Sec. 7.3.2), this creates interactive scenarios where the EV must show diverse strategies to succeed, such as nudging into the lane and waiting for the neighboring vehicle to break and create a gap.
Not having found a single interactive lane change, we extend the search to all scenarios tagged with lane changes where PDM-Closed achieves a score below 90%. In all cases, the EV either gets stuck in a pickup/dropoff zone, or the bad score stems from uncomfortable driving or slow progress. Although the expert does change lanes, this is not required to follow the route. In contrast, *interPlan*'s LTD, MTD, and HTD scenarios enforce lane changes via the navigation route and explicitly reward them in the score.

7.4 Results

7.4.1 Generalization of State-of-the-Art Planning Methods

Our experimental results are shown in Tab. 7.3. We find that PDM-Closed achieves the best results with an overall score of 42%. However, in contrast to previous results on the Val14 test-split and the nuPlan leaderboard, it is far from achieving a perfect score. In particular, it achieves the best score among all state-of-the-art planners in the Nudge, and Overtake scenarios. This is mainly due to its ability to sample different lateral offsets from the centerline it is following, enabling it to pass some of the obstacles in the lane. Moreover, it outperforms all other methods in the MTD lane change scenarios. However, due to its limitation of not being able to do lane changes, this merely reflects its ability to find an optimal trajectory concerning comfort and safety since it is unable to reach the goal lane in these scenarios. In addition, almost all methods in our benchmark fail in the Acc. and Overt. scenarios, resulting in a score of 0%. Exceptions are DTPP for Acc. and PDM-Open and PDM-Closed for the Overt. scenarios. While DTPP successfully makes a lane change before the stationary vehicles in the Acc. scenario shown in Fig. 7.2a, PDM-Closed stops before them (see 7.2e). Similarly, PDM-Open and PDM-Closed pass the parked car in the easiest of the 10 overtake scenarios, i.e., the one with no oncoming traffic. For PDM-Open, this is not intentional but instead caused by a failure to follow the curved centerline since this model does not consider surrounding vehicles. Surprisingly, PDM-Closed performs worse than IDM in the scenarios with Jaywalkers even though it employs an IDM policy for centerline following. In contrast to IDM, PDM uses a future horizon of only 2.0s to evaluate potential longitudinal profiles. In two scenarios, a collision is unavoidable when the pedestrian is inside this horizon. In a further scenario, PDM-Closed avoids a collision by evading the pedestrians at high speed (see Fig. 7.2f instead of stopping and waiting for them to cross (see Fig. 7.2b). Overall, our results reveal critical shortcomings in the generalization capabilities of state-of-the-art planning methods.

7.4.2 Interactive Lane Changes

In the lane change scenarios (i.e., LTD, MTD, HTD), we often observe trajectories that result in collisions or violations of the drivable area for the learning-based planners UrbanDriver, GameFormer, and PDM-Open (see Fig. 7.2h). GC-PGP often stops and gets stuck (see Fig. 7.2d). On the other hand, rule-based centerline planners (PDM-Closed, IDM) do not cause these infractions and thus incur smaller penalties. However, they do not reach the goal lane even once. MOBIL extends IDM with a criterion to achieve safe lane changes, depending on the surroundings. We use a simple implementation that, given that MOBIL approves the lane change, tasks the controller with reaching waypoints on the neighboring lane. It achieves the highest rate of reaching the goal lane at 52%. However, this comes at the cost of some drivable area infractions and collisions. We

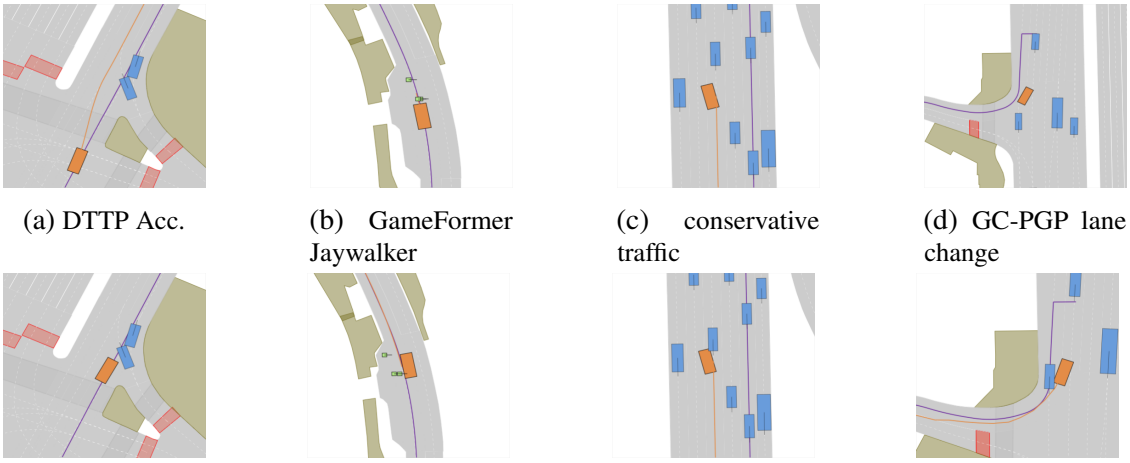
find that some collisions are explained by the downstream controller not being able to merge to the neighboring lane without oscillating, and others are caused by erroneously expecting vehicles approaching from behind to break (see Fig. 7.2g). Nonetheless, it has the highest rate of reaching the goal lane and performs well if the surrounding vehicles act conservatively (see Fig. 7.2c). We conclude that most state-of-the-art planners lack sophisticated lane change trajectory planning, such as synchronizing to a gap in preparation, proactively influencing other vehicles' behavior, and aborting a lane change when necessary.

7.4.3 Rule-Based vs. Learning-Based Planning

Notably, the rule-based planners IDM, IDM+MOBIL, and PDM-Closed outperform the learning-based contestants. This is especially surprising for IDM and PDM-Closed, since they are by design limited to simple lane-following without lane changes or overtakes and thus unable to solve scenarios where a construction zone, a parked vehicle, or an accident blocks the current lane. Moreover, the lowest scores are achieved by planners that regress waypoints (GC-PGP, UrbanDriver) instead of selecting a trajectory among a set of feasible samples. This indicates that they have the lowest generalization capabilities and are susceptible even to small distributional shifts. The best learning-based planner is DTPP, which uses a learned cost function alongside trajectories generated by a sampler. Overall, none of the models demonstrates sufficient world understanding to navigate successfully through these difficult scenarios.

7.5 LLM-Based Planning

In the previous section, we assessed state-of-the-art planning methods under difficult conditions and revealed that there is a critical lack of generalization ability to solve rare and difficult scenarios. With the rise of foundation models and the tremendous interest surrounding them recently, exploiting their impressive world understanding and generalization capabilities in the field of autonomous driving became a vital field of research. As the scenarios we proposed in our benchmark demand a thorough understanding of traffic scenarios and complex reasoning (such as anticipating a pedestrian to unlawfully cross the street at a bus stop), approaching the planning task with foundation models appears to be a promising strategy. Prior work achieves impressive results using LLMs for open-loop trajectory planning in real-world driving scenarios. However, the current state of the art on regular nuPlan scenarios suggests that such common scenarios can be solved efficiently with simple rule-based planners, and the promise of better generalization to long-tail scenarios is yet to be put to test. In the following, we explore the capabilities of LLMs in the context of closed-loop behavior planning in such scenarios. First, we establish a simple baseline, which is an adaptation of the open-loop GPT-Driver (Mao *et al.*, 2023a) planner. Then, we describe our LLM-based behavior planning method. We evaluate



(a) DTTP Acc. (b) GameFormer Jaywalker (c) conservative traffic (d) GC-PGP lane change

(e) PDM-Closed Acc. (f) PDM-Closed Jaywalker (g) assertive traffic (h) UrbanDriver lane change

Figure 7.2: **Qualitative Results.** Fig. (a), (b), (c) show successfully avoiding an accident site, waiting for jaywalkers, and executing a lane change surrounded by conservative agents. Failure cases are stopping before the accident site (e), narrowly avoiding the pedestrians (f), and colliding with an assertive vehicle approaching quickly from behind (g). Fig. (d) and (h) show failures of learning-based planners, such as suddenly stopping (d) or causing a collision (h). The planned trajectory is depicted in orange, the navigation route in purple.

both methods on the same *interPlan* benchmark we used for the state-of-the-art planning methods. By releasing our code, we hope to facilitate further research on improved LLM-based planning for Automated Driving (AD).

7.5.1 LLM Waypoints Planning

GPTDriver (Mao *et al.*, 2023a) fine-tunes and instructs an LLM to predict a set of future waypoints. The prompt comprises general task instruction, perception context, and ego-states. Chain-of-thought reasoning is used to improve the interpretability and quality of the output. We adapt the method to the nuPlan requirements by changing the output to an 8-second trajectory at 2 Hz and by including route information into the prompt. Thus, we provide the planner with a 'Mission Goal' telling it which lanes are on route. We test various LLMs, including Llama-7B, Llama-13B (Touvron *et al.*, 2023), and GPT-3.5. We call this baseline `LLM-WP` and fine-tune it on a set of 600 diverse and unaugmented nuPlan scenarios for 3 epochs. For Llama we employ QLoRa (Dettmers *et al.*, 2024) using a rank of 64 and a learning rate of 2e-4. For GPT, we use openAI's fine-tuning API.

Model	LLM	interPlan	Constr.	Acc.	Jayw.	Nudge	Overt.	LTD	MTD	HTD	Driv.	Goal	No-Col.
PDM-Closed	None	42	18	0	**48**	74	9	62	62	62	**100**	0	**100**
LLM-Hybrid	LLama-7B	**53**	27	**20**	**48**	**93**	**28**	**81**	48	**80**	**100**	43	87
LLM-Hybrid	LLama-13B	48	36	10	**48**	82	19	71	40	77	97	43	83
LLM-Hybrid	GPT-3.5	40	**52**	0	**48**	25	9	76	34	69	97	**47**	77
LLM-WP	LLama-7B	16	0	0	0	0	0	29	**64**	37	90	0	67
LLM-WP	LLama-13B	17	0	0	0	0	0	31	**64**	38	87	0	70
LLM-WP	GPT-3.5	22	0	0	0	0	0	64	41	69	93	0	**100**

Table 7.4: **LLM baselines on the interPlan benchmark.**

7.5.2 LLM Behavior Planning

In addition, we propose a two-stage hybrid planner that combines an LLM behavior planner with a rule-based motion planner. In the first stage, the traffic scenario is described in a natural language prompt. As in the case of our LLM-WP baseline, it comprises general task instruction, perception context, and ego states, as well as route information. Finally, the model is instructed to select a suitable behavior from a list of available behaviors. We define the following behaviors: follow the current lane, merge left, merge right, overtake obstacle, stop and wait. We filter the list based on the availability of neighboring lanes and the presence of obstacles in the current lane before presenting the options to the LLM. Each option is associated with a centerline and an offset from it. For instance, 'overtake obstacle' refers to the current centerline and the offset required to pass the obstacle, whereas 'merge left' refers to the left neighboring centerline with no offset. The second stage applies a rule-based motion planner to find an optimal trajectory conditioned on the selected behavior parameters. We leverage PDM-Closed (Dauner *et al.*, 2023), the top-performing method in our benchmark, for motion planning. This algorithm samples lateral offsets and longitudinal speed profiles locally around the chosen behavior and selects a trajectory based on a cost function. The motion planner runs at 10 Hz, while the LLM behavior planner is queried at 1 Hz. We employ the LLM in a zero-shot setting, as we want to test the generalization of planning methods to unseen difficult scenarios. Moreover, labeling the regular nuPlan scenarios would almost exclusively result in 'follow current lane' samples. We call this method LLM-Hybrid.

7.5.3 Rule-Based Motion Planners can be enhanced with LLMs

Tab. 7.4 compares the results of both LLM-based planning methods to the best method on our benchmark, i.e., PDM-Closed. Despite using the same prompt and fine-tuning strategy, the LLM-WP baseline cannot reproduce the strong results from other benchmarks (see (Mao *et al.*, 2023a)) and trivially outputs a longitudinal profile at a constant heading angle. Conversely, our LLM-Hybrid outperforms PDM-Closed, setting a new state-of-the-art on our proposed benchmark. Surprisingly, the larger 13B Llama model performs worse than the smaller 7B version, often making questionable decisions, resulting in collisions or less progress.

7.5.4 LLMs lack Traffic Understanding

While the `LLM-Hybrid` outperforms all existing methods, it is merely a strong baseline and unable to solve all the difficult *interPlan* scenarios. For instance, we observe that it often toggles between different behaviors, such as overtaking and waiting for oncoming traffic, ultimately getting stuck behind the obstacle. Moreover, the decision is often inconsistent with the situation analysis, indicating a lack of understanding of traffic. Thus, we highlight fine-tuning with auxiliary tasks to enhance traffic understanding as an important direction for future research.

7.6 Conclusion

This chapter presented a novel realistic benchmark that evaluates planning algorithms in highly interactive and rare long-tail scenarios. We found that existing state-of-the-art methods exhibit critical limitations in generalizing to such scenarios. Moreover, we provided an LLM-based planning baseline, proposed a novel LLM-based behavior planner, and explored their capabilities.

Chapter 8

Conclusion & Outlook

This dissertation delved into several aspects of the autonomous driving task. In particular, we explored and addressed several problems in vehicle trajectory prediction and motion planning. In general, we present contributions to these fields that aim to enable the development of autonomous systems that can handle urban driving scenarios, which are characterized by diverse road topologies, complex interactions with other agents, high safety requirements, and unforeseen events.

In Chapter 2, we laid out the foundations of both tasks and defined them formally. Then, we provided an exhaustive overview of state-of-the-art methods, prevalent paradigms, and technical aspects, including input and output representations, as well as common model architectures. Moreover, we argued that considering prediction and planning as tightly coupled and interconnected tasks is paramount to making safe and confident decisions in interactive traffic scenarios. Consequently, in an effort to bring together experts from both fields and discuss suitable interfaces, benchmarks, and system designs, we organized the *IROS 2024 Workshop on interaction-aware autonomous systems* as well as the *Workshop on interaction-driven behavior prediction and motion planning* at *IV2024* and *ITSC2024*.

Then, in chapter 3 we started by exploring latent space distributions of CVAE models for trajectory prediction. We found that simplistic gaussian latent spaces are not expressive enough for this complex task and proposed to leverage rich GMM latent spaces. By combining them with the Unscented Transform, we were able to estimate a parametric output distribution. Moreover, we showcased that this estimate is better than using random samples, while at the same time being deterministic and reproducible, which is a major requirement for safety-critical system components.

Another crucial requirement set to prediction methods is generalization. Hence, in chapter 4, we address the aspect of generalizing to diverse map layouts. We unveil that previously discovered shortcomings of state-of-the-art methods are caused by insufficient map understanding. To counter this, we propose to leverage scene representations based on Frenet frames along lane centerlines. We present a general wrapper that is compatible with many methods and demonstrate that it can reduce the rate of inadmissible off-road predictions in difficult scenarios by an order of magnitude.

As chapters 3 and 4 presented advances in the prediction subtask, we then demonstrate how to carry them over to the downstream trajectory planning tasks. To this end, chapter 5

highlights the similarities between both tasks. In particular, both tasks share a common input representation and model architectures. However, we identified goal-conditioning as a key difference. Therefore, we propose a novel post-conditioning method that can be applied to predictors without retraining. With that, we can repurpose powerful trajectory prediction models for motion planning. We demonstrate this by applying it to a state-of-the-art predictor and obtain a strong open-loop planning baseline named GC-PGP.

Chapter 6 builds on this and presents a rule-based planner named PDM-Closed. It extends the well-known IDM policy with a simplistic prediction module and an interpretable cost function. Benchmarking this against several rule-based and learning-based planning methods revealed several surprising findings. In particular, we found that, against common conception, rule-based methods can outperform sophisticated learning-based contestants. Moreover, we explored a misalignment between open-loop and closed-loop benchmarks, the two main evaluation modes for planning methods. Finally, we present a hybrid planner that combines PDM-Closed with a learning-based open-loop planning method and achieves state-of-the-art performance on the nuPlan benchmark. A version of this planner that combined the rule-based PDM-Closed with GC-PGP, presented in chapter 5 claimed victory in the *2023 nuPlan driving competition*.

Fueled by the impressive performance of the PDM-Closed planner, we ask whether this method can take on the challenge of handling real-world traffic. In chapter 7, we argue that the nuPlan benchmark does not cover generalization to long-tail scenarios or highly interactive maneuvers. Therefore, we propose a novel benchmark based on these scenarios that is tailored to test the generalization capabilities of planning methods. We evaluate a comprehensive set of planning methods and reveal shortcomings in the current state-of-the-art. We hope this provides future research with a clear picture of relevant problems that must be solved before these methods can be applied in the real world.

Symbols

t_0	Current time point
t_f	Future prediction / planning horizon
t_h	Past observation horizon
$s_t^{(v)}$	State of vehicle v at time t
X_v	Past trajectory of vehicle v
Y_v	Future trajectory of vehicle v
X_{EV}	Past trajectory of the ego vehicle
Y_{EV}	Future trajectory of the ego vehicle
$\overline{\mathbf{X}}_{\mathrm{SV}}$	Past trajectories of all surrounding vehicles
$\mathcal{F}^{(i)}$	Frenet Frame defined by centerline i
$x_t^{(a)}$	x coordinate of agent a at time t
$y_t^{(a)}$	y coordinate of agent a at time t
$v_t^{(a)}$	speed of agent a at time t
$a_t^{(a)}$	acceleration of agent a at time t
$\omega_t^{(a)}$	yaw rate of agent a at time t

Abbreviations

AD	Automated Driving
ADE	Average Displacement Error
ADS	automated driving system
BC	Behavior Cloning
BEV	Bird's Eye View
CA	Constant Acceleration Model
CLS-R	nuPlan closed-loop score with reactive traffic policy
CLS-NR	nuPlan closed-loop score with non-reactive traffic policy
CNN	Convolutional Neural Network
CUAE	Conditional Unscented Autoencoder
CV	Constant Velocity Model
CVAE	Conditional Variational Autoencoder
CXP	conditional ex-post
DL	Deep Learning
DO	Ablation with dropouts on historical states in input features
DNN	Deep Neural Networks
ELBO	Evidence Lower Bound
E2E	end-to-end
EV	ego vehicle
FDE	Final Displacement Error
GC-PGP	Goal-Conditioned Planning with Graph-based Policy
GAN	Generative Adversarial Network
GAT	Graph Attention Network
GMM	Gaussian Mixture Model
GNN	Graph Neural Network
GPS	Global Positioning System
GRU	Gated Recurrent Unit
HTD	High traffic density
IDM	Intelligent Driver Model
IL	Imitation Learning
IPPS	Integrated Prediction and Planning System
KF	Kalman Filters
KL	Kullback-Leibler
LLM	Large Language Model

LSTM	Long Short-Term Memory
LTD	Low traffic density
minADE	Minimum Average Displacement Error
minFDE	Minimum Final Displacement Error
MAE	Mean Absolute Error
MHA	Multi-Head Attention
MSE	Mean Squared Error
MTD	Medium traffic density
MIED	Mean Inter-Endpoint Distance
MLP	Multilayer Perceptron
MR	Miss Rate
NF	Normalizing Flow
NH	Ablation with no historical states in input features
NN	Neural Network
NLL	Negative Log Likelihood
ORP	Offroad Probability
OLS	nuPlan open-loop score
PDM	Predictive Driver Model
PDM-Open	Predictive Driver Model for open-loop driving
PDM-Closed	Predictive Driver Model for closed-loop driving
PDM-Hybrid	Predictive Driver Model for open-loop and closed-loop driving
PS	Planning System
RNN	Recurrent Neural Networks
RL	Reinforcement Learning
SotA	State-of-the-Art
SV	surrounding vehicle
TPI	Temporal Plan Inconsistency
TV	target vehicle
UAE	Unscented Autoencoder
UD-X	Downscaled Urban Driver model with X parameters
UT	Unscented Transform
VAE	Variational Autoencoders
WTA	winner-takes-all

Contributions Overview

This chapter provides a detailed overview of the contributions made to joint works conducted as part of this dissertation and presented in this document. For each paper, a separate table clearly delineates the specific roles and contributions of all co-authors.

Author	Author position	Scientific ideas [%]	Data generation [%]	Analysis interpretation [%]	Paper writing
Steffen Hagedorn	1*	40	40	40	40
Marcel Hallgarten	2*	40	40	40	40
Martin Stoll	3	10	10	10	10
Alexandru Paul Condurache	4	10	10	10	10
Title of paper		The Integration of Prediction and Planning in Deep Learning Automated Driving Systems: A Review			
Status of Publication		published			

Table 1: **Contribution Overview for Chapter 2**

Author	Author position	Scientific ideas [%]	Data generation [%]	Analysis interpretation [%]	Paper writing
Faris Janjoš	1	40	30	35	40
Marcel Hallgarten	2	30	40	35	40
Anthony Knittel	3	15	15	15	5
Maxim Dolgov	4	5	5	5	5
Andreas Zell	5	5	5	5	5
J. Marius Zöllner	6	5	5	5	5
Title of paper		Conditional Unscented Autoencoders for Trajectory Prediction			
Status of Publication		accepted			

Table 2: **Contribution Overview for Chapter 3**

Author	Author position	Scientific ideas [%]	Data generation [%]	Analysis interpretation [%]	Paper writing
Marcel Hallgarten	1	80	70	80	85
Ismail Kisa	2	10	20	10	5
Martin Stoll	3	5	5	5	5
Andreas Zell	4	5	5	5	5
Title of paper		Stay on Track: A Frenet Wrapper to Overcome Off-road Trajectories in Vehicle Motion Prediction			
Status of Publication		published			

Table 3: **Contribution Overview for Chapter 4**

Author	Author position	Scientific ideas [%]	Data generation [%]	Analysis interpretation [%]	Paper writing
Marcel Hallgarten	1	80	80	80	80
Martin Stoll	2	10	10	10	10
Andreas Zell	3	10	10	10	10
Title of paper		From Prediction to Planning with Goal Conditioned Lane Graph Traversals			
Status of Publication		published			

Table 4: **Contribution Overview for Chapter 5**

Author	Author position	Scientific ideas [%]	Data generation [%]	Analysis interpretation [%]	Paper writing
Daniel Dauner	1	40	40	40	40
Marcel Hallgarten	2	30	30	30	30
Andreas Geiger	3	10	10	10	10
Kashyap Chitta	4	20	20	20	20
Title of paper		Parting with Misconceptions about Learning-based Vehicle Motion Planning			
Status of Publication		published			

Table 5: **Contribution Overview for Chapter 6**

Author	Author position	Scientific ideas [%]	Data generation [%]	Analysis interpretation [%]	Paper writing
Marcel Hallgarten	1	70	70	70	70
Julian Zapata	2	15	15	15	15
Martin Stoll	3	5	5	5	5
Katrin Renz	4	5	5	5	5
Andreas Zell	5	5	5	5	5
Title of paper		Can Vehicle Motion Planning Generalize to Realistic Long-Tail Scenarios?			
Status of Publication		published			

Table 6: **Contribution Overview for Chapter 7**

Author	Author position	Scientific ideas [%]	Data generation [%]	Analysis interpretation [%]	Paper writing
Daniel Dauner	1	40	40	40	40
Marcel Hallgarten	2	30	30	30	30
Tinayu Li	3	5	5	2	1
Xinshuo Weng	4	0	3	1	1
Zhiyu Huang	5	0	2	1	1
Zetong Yang	6	0	0	1	1
Hongyang Li	7	0	0	1	1
Igor Gilitschenski	8	0	0	1	1
Boris Ivanovic	9	0	0	1	1
Marco Pavone	10	0	0	1	1
Andreas Geiger	11	5	0	1	2
Kashyap Chitta	12	20	20	20	20
Title of paper		NAVSIM: Data-Driven Non-Reactive Autonomous Vehicle Simulation and Benchmarking			
Status of Publication		published			

Table 7: **Contribution Overview (paper not covered in dissertation)**

Bibliography

Abeysirigoonawardena, Y., Shkurti, F., and Dudek, G. (2019). Generating adversarial driving scenarios in high-fidelity simulators. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8271–8277. IEEE.

Aghasadeghi, N. and Bretl, T. (2011). Maximum entropy inverse reinforcement learning in continuous state spaces with path integrals. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1561–1566. IEEE.

Agro, B., Sykora, Q., Casas, S., and Urtasun, R. (2023). Implicit occupancy flow fields for perception and prediction in self-driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1379–1388.

Ajanovic, Z., Lacevic, B., Shyrokau, B., Stolz, M., and Horn, M. (2018). Search-based optimal motion planning for automated driving. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4523–4530. IEEE.

Albeaik, S., Bayen, A., Chiri, M. T., Gong, X., Hayat, A., Kardous, N., Keimer, A., McQuade, S. T., Piccoli, B., and You, Y. (2022). Limitations and improvements of the intelligent driver model (idm). *SIAM Journal on Applied Dynamical Systems*, **21**(3), 1862–1892.

Althoff, M. and Lutz, S. (2018). Automatic generation of safety-critical test scenarios for collision avoidance of road vehicles. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1326–1333. IEEE.

Althoff, M., Koschi, M., and Manzinger, S. (2017). Commonroad: Composable benchmarks for motion planning on roads. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 719–726. IEEE.

Amirloo, E., Rasouli, A., Lakner, P., Rohani, M., and Luo, J. (2022). Latentformer: Multi-agent transformer-based interaction modeling and trajectory prediction. *arXiv preprint arXiv:2203.01880*.

Arjovsky, M. and Bottou, L. (2017). Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862*.

Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *International conference on machine learning*. PMLR.

Arnab, A., Dehghani, M., Heigold, G., Sun, C., Lučić, M., and Schmid, C. (2021). Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6836–6846.

Bacha, A., Bauman, C., Faruque, R., Fleming, M., Terwelp, C., Reinholtz, C., Hong, D., Wicks, A., Alberi, T., Anderson, D., *et al.* (2008). Odin: Team victortango's entry in the darpa urban challenge. *Journal of field Robotics*, **25**(8), 467–492.

Bae, S., Isele, D., Nakhaei, A., Xu, P., Anon, A. M., Choi, C., Fujimura, K., and Moura, S. (2022). Lane-change in dense traffic with model predictive control and neural networks. *IEEE Transactions on Control Systems Technology*, **31**(2), 646–659.

Bagschik, G., Menzel, T., and Maurer, M. (2018). Ontology based scene creation for the development of automated vehicles. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1813–1820. IEEE.

Bahari, M., Saadatnejad, S., Rahimi, A., Shaverdikondori, M., Shahidzadeh, A. H., Moosavi-Dezfooli, S.-M., and Alahi, A. (2022). Vehicle trajectory prediction works, but not everywhere. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17123–17133.

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Bain, M. and Sammut, C. (1995). A framework for behavioural cloning. In *Machine Intelligence 15*, pages 103–129.

Bajcsy, A., Siththaranjan, A., Tomlin, C. J., and Dragan, A. D. (2021). Analyzing human models that adapt online. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2754–2760. IEEE.

Bandyopadhyay, T., Won, K. S., Frazzoli, E., Hsu, D., Lee, W. S., and Rus, D. (2013). Intention-aware motion planning. In *Algorithmic Foundations of Robotics X: Proceedings of the Tenth Workshop on the Algorithmic Foundations of Robotics*, pages 475–491. Springer.

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

Behl, A., Chitta, K., Prakash, A., Ohn-Bar, E., and Geiger, A. (2020). Label efficient visual abstractions for autonomous driving. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2338–2345. IEEE.

Bergamini, L., Ye, Y., Scheel, O., Chen, L., Hu, C., Del Pero, L., Osiński, B., Grimmett, H., and Ondruska, P. (2021). Simnet: Learning reactive self-driving simulations from

real-world observations. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5119–5125. IEEE.

Bhattacharyya, P., Huang, C., and Czarnecki, K. (2024). Ssl-interactions: Pretext tasks for interactive trajectory prediction. *arXiv:2401.07729*.

Birkemeyer, L., Pett, T., Vogelsang, A., Seidl, C., and Schaefer, I. (2022). Feature-interaction sampling for scenario-based testing of advanced driver assistance systems*. In *Proceedings of the 16th International Working Conference on Variability Modelling of Software-Intensive Systems*, pages 1–10.

Biswas, S., Casas, S., Sykora, Q., Agro, B., Sadat, A., and Urtasun, R. (2024). Quad: Query-based interpretable neural motion planning for autonomous driving. *arXiv:2404.01486*.

Bock, J., Krajewski, R., Moers, T., Runde, S., Vater, L., and Eckstein, L. (2020). The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 1929–1934. IEEE.

Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., *et al.* (2016). End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*.

Boloor, A., Garimella, K., He, X., Gill, C., Vorobeychik, Y., and Zhang, X. (2020). Attacking vision-based perception in end-to-end autonomous driving models. *Journal of Systems Architecture*, **110**, 101766.

Boulton, F. A., Grigore, E. C., and Wolff, E. M. (2020). Motion prediction using trajectory sets and self-driving domain knowledge. *arXiv:2006.04767*.

Bronstein, E., Palatucci, M., Notz, D., White, B., Kuefler, A., Lu, Y., Paul, S., Nikdel, P., Mougin, P., Chen, H., *et al.* (2022). Hierarchical model-based imitation learning for planning in autonomous driving. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8652–8659. IEEE.

Buehler, M., Iagnemma, K., and Singh, S. (2009). *The DARPA urban challenge: autonomous vehicles in city traffic*, volume 56. springer.

Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., and Beijbom, O. (2020). nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631.

Caesar, H., Kabzan, J., Tan, K. S., Fong, W. K., Wolff, E., Lang, A., Fletcher, L., Beijbom, O., and Omari, S. (2021). nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles. *arXiv preprint arXiv:2106.11810*.

Casas, S., Luo, W., and Urtasun, R. (2018). Intentnet: Learning to predict intention from raw sensor data. In *Conference on Robot Learning*, pages 947–956. PMLR.

Casas, S., Gulino, C., Suo, S., Luo, K., Liao, R., and Urtasun, R. (2020a). Implicit latent variable model for scene-consistent motion forecasting. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIII 16*, pages 624–641. Springer.

Casas, S., Gulino, C., Suo, S., and Urtasun, R. (2020b). The importance of prior knowledge in precise multimodal prediction. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2295–2302. IEEE.

Casas, S., Gulino, C., Liao, R., and Urtasun, R. (2020c). Spagnn: Spatially-aware graph neural networks for relational behavior forecasting from sensor data. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9491–9497. IEEE.

Casas, S., Sadat, A., and Urtasun, R. (2021). Mp3: A unified model to map, perceive, predict and plan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14403–14412.

Chai, Y., Sapp, B., Bansal, M., and Anguelov, D. (2019). Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. *arXiv preprint arXiv:1910.05449*.

Chang, M.-F., Lambert, J., Sangkloy, P., Singh, J., Bak, S., Hartnett, A., Wang, D., Carr, P., Lucey, S., Ramanan, D., *et al.* (2019). Argoverse: 3d tracking and forecasting with rich maps. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8748–8757.

Chekroun, R., Gilles, T., Toromanoff, M., Hornauer, S., and Moutarde, F. (2023). Mbappe: Mcts-built-around prediction for planning explicitly. *arXiv preprint arXiv:2309.08452*.

Chen, C., Seff, A., Kornhauser, A., and Xiao, J. (2015). Deepdriving: Learning affordance for direct perception in autonomous driving. In *Proceedings of the IEEE international conference on computer vision*, pages 2722–2730.

Chen, D. and Krähenbühl, P. (2022). Learning from all vehicles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17222–17231.

Chen, D., Zhou, B., Koltun, V., and Krähenbühl, P. (2020). Learning by cheating. In *Conference on Robot Learning*, pages 66–75. PMLR.

Chen, J., Yuan, B., and Tomizuka, M. (2019). Deep imitation learning for autonomous driving in generic urban scenarios with enhanced safety. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2884–2890. IEEE.

Chen, L., Sinavski, O., Hünermann, J., Karnsund, A., Willmott, A. J., Birch, D., Maund, D., and Shotton, J. (2023a). Driving with llms: Fusing object-level vector modality for explainable autonomous driving. *arXiv preprint arXiv:2310.01957*.

Chen, L., Wu, P., Chitta, K., Jaeger, B., Geiger, A., and Li, H. (2023b). End-to-end autonomous driving: Challenges and frontiers. *arXiv preprint arXiv:2306.16927*.

Chen, S., Jiang, B., Gao, H., Liao, B., Xu, Q., Zhang, Q., Huang, C., Liu, W., and Wang, X. (2024). Vadv2: End-to-end vectorized autonomous driving via probabilistic planning. *arXiv:2402.13243*.

Chen, W., Wang, F., and Sun, H. (2021). S2tnet: Spatio-temporal transformer networks for trajectory prediction in autonomous driving. In *Asian Conference on Machine Learning*, pages 454–469. PMLR.

Chen, Y., Ivanovic, B., and Pavone, M. (2022). Scept: Scene-consistent, policy-based trajectory predictions for planning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17103–17112.

Chen, Y., Tonkens, S., and Pavone, M. (2023c). Categorical traffic transformer: Interpretable and diverse behavior prediction with tokenized latent. *arXiv:2311.18307*.

Chen, Y., Veer, S., Karkus, P., and Pavone, M. (2023d). Interactive joint planning for autonomous vehicles. *IEEE RA-L*.

Chen, Y., Karkus, P., Ivanovic, B., Weng, X., and Pavone, M. (2023e). Tree-structured policy planning with learned behavior models. *arXiv preprint arXiv:2301.11902*.

Chen, Z., Ye, M., Xu, S., Cao, T., and Chen, Q. (2023f). Deepemplanner: An em motion planner with iterative interactions. *arXiv:2311.08100*.

Cheng, J., Chen, Y., Mei, X., Yang, B., Li, B., and Liu, M. (2023). Rethinking imitation-based planner for autonomous driving. *arXiv preprint arXiv:2309.10443*.

Cheng, J., Chen, Y., and Chen, Q. (2024). Pluto: Pushing the limit of imitation learning-based planning for autonomous driving. *arXiv:2404.14327*.

Chitta, K., Prakash, A., and Geiger, A. (2021). Neat: Neural attention fields for end-to-end autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15793–15803.

Chitta, K., Prakash, A., Jaeger, B., Yu, Z., Renz, K., and Geiger, A. (2022). Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Choi, C., Malla, S., Patil, A., and Choi, J. H. (2021). Drogon: A trajectory prediction model based on intention-conditioned behavior reasoning. In *Conference on Robot Learning*, pages 49–63. PMLR.

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Claussmann, L., Revilloud, M., Gruyer, D., and Glaser, S. (2019). A review of motion planning for highway autonomous driving. *IEEE Transactions on Intelligent Transportation Systems*, **21**(5), 1826–1848.

Codevilla, F., Müller, M., López, A., Koltun, V., and Dosovitskiy, A. (2018a). End-to-end driving via conditional imitation learning. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 4693–4700. IEEE.

Codevilla, F., Lopez, A. M., Koltun, V., and Dosovitskiy, A. (2018b). On offline evaluation of vision-based driving models. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 236–251.

Codevilla, F., Santana, E., López, A. M., and Gaidon, A. (2019). Exploring the limitations of behavior cloning for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9329–9338.

Costa-Gomes, M. A., Crawford, V. P., and Iriberri, N. (2009). Comparing models of strategic thinking in van huyck, battalio, and beil's coordination games. *Journal of the European Economic Association*, **7**(2-3), 365–376.

Cui, A., Casas, S., Sadat, A., Liao, R., and Urtasun, R. (2021). Lookout: Diverse multi-future prediction and planning for self-driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16107–16116.

Cui, A., Casas, S., Wong, K., Suo, S., and Urtasun, R. (2022). Gorela: Go relative for viewpoint-invariant motion forecasting. *arXiv preprint arXiv:2211.02545*.

Cui, A., Casas, S., Wong, K., Suo, S., and Urtasun, R. (2023). Gorela: Go relative for viewpoint-invariant motion forecasting. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE.

Cui, H., Radosavljevic, V., Chou, F.-C., Lin, T.-H., Nguyen, T., Huang, T.-K., Schneider, J., and Djuric, N. (2019). Multimodal trajectory predictions for autonomous driving using deep convolutional networks. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 2090–2096. IEEE.

Cui, H., Nguyen, T., Chou, F.-C., Lin, T.-H., Schneider, J., Bradley, D., and Djuric, N. (2020). Deep kinematic models for kinematically feasible vehicle trajectory predictions. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE.

Cunningham, A. G., Galceran, E., Eustice, R. M., and Olson, E. (2015). Mpdm: Multipolicy decision-making in dynamic, uncertain environments for autonomous driving. In *IEEE ICRA*, pages 1670–1677.

Daftry, S., Bagnell, J. A., and Hebert, M. (2017). Learning transferable policies for monocular reactive mav control. In *2016 International Symposium on Experimental Robotics*, pages 3–11. Springer.

Danesh, M. H., Cai, P., and Hsu, D. (2023). Leader: Learning attention over driving behaviors for planning under uncertainty. In *Conference on robot learning*, pages 199–211. PMLR.

Dauner, D., Hallgarten, M., Geiger, A., and Chitta, K. (2023). Parting with misconceptions about learning-based vehicle motion planning. *arXiv preprint arXiv:2306.07962*.

Dauner, D., Hallgarten, M., Li, T., Weng, X., Huang, Z., Yang, Z., Li, H., Gilitschenski, I., Ivanovic, B., Pavone, M., *et al.* (2024). Navsim: Data-driven non-reactive autonomous vehicle simulation and benchmarking. *arXiv:2406.15349*.

Demmler, T., Tamke, A., Dang, T., Haug, K., and Mikelsons, L. (2024). Towards consistent and explainable motion prediction using heterogeneous graph attention. *arXiv:2405.10134*.

Dendorfer, P., Osep, A., and Leal-Taixé, L. (2020). Goal-gan: Multimodal trajectory prediction based on goal position estimation. In *Proceedings of the Asian Conference on Computer Vision*.

Dendorfer, P., Elflein, S., and Leal-Taixé, L. (2021). Mg-gan: A multi-generator model preventing out-of-distribution samples in pedestrian trajectory prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*.

Deo, N. and Trivedi, M. M. (2020). Trajectory forecasts in unknown environments conditioned on grid-based plans. *arXiv preprint arXiv:2001.00735*.

Deo, N., Wolff, E., and Beijbom, O. (2022). Multimodal trajectory prediction conditioned on lane-graph traversals. In *Conference on Robot Learning*, pages 203–212. PMLR.

Derbel, O., Peter, T., Zebiri, H., Mourllion, B., and Basset, M. (2013). Modified intelligent driver model for driver safety and traffic stability improvement. *IFAC Proceedings Volumes*, **46**(21), 744–749.

Dettmers, T., Pagnoni, A., Holtzman, A., and Zettlemoyer, L. (2024). Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, **36**.

Diehl, F., Brunner, T., Le, M. T., and Knoll, A. (2019). Graph neural networks for modelling traffic participant interaction. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 695–701. IEEE.

Ding, W., Chen, B., Xu, M., and Zhao, D. (2020). Learning to collide: An adaptive safety-critical scenarios generating method. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2243–2250. IEEE.

Ding, W., Zhang, L., Chen, J., and Shen, S. (2021a). Epsilon: An efficient planning system for automated vehicles in highly interactive environments. *IEEE Transactions on Robotics*, **38**(2), 1118–1138.

Ding, W., Chen, B., Li, B., Eun, K. J., and Zhao, D. (2021b). Multimodal safety-critical scenarios generation for decision-making algorithms evaluation. *IEEE Robotics and Automation Letters*, **6**(2), 1551–1558.

Ding, Z. and Zhao, H. (2023). Incorporating driving knowledge in deep learning based vehicle trajectory prediction: A survey. *IEEE Transactions on Intelligent Vehicles*.

Djuric, N., Radosavljevic, V., Cui, H., Nguyen, T., Chou, F.-C., Lin, T.-H., and Schneider, J. (2018). Short-term motion prediction of traffic actors for autonomous driving using deep convolutional networks. *arXiv preprint arXiv:1808.05819*, **1**(2), 6.

Do, M. N. (2003). Fast approximation of kullback-leibler distance for dependence trees and hidden markov models. *IEEE signal processing letters*, **10**(4), 115–118.

Dong, S., Wang, P., and Abbas, K. (2021). A survey on deep learning and its applications. *Computer Science Review*, **40**, 100379.

Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., and Koltun, V. (2017). Carla: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR.

Erz, J., Schütt, B., Braun, T., Guissouma, H., and Sax, E. (2022). Towards an ontology that reconciles the operational design domain, scenario-based testing, and automated vehicle architectures. In *2022 IEEE international systems conference (SYSCON)*, pages 1–8. IEEE.

Ettinger, S., Cheng, S., Caine, B., Liu, C., Zhao, H., Pradhan, S., Chai, Y., Sapp, B., Qi, C. R., Zhou, Y., *et al.* (2021). Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9710–9719.

Fan, H., Zhu, F., Liu, C., Zhang, L., Zhuang, L., Li, D., Zhu, W., Hu, J., Li, H., and Kong, Q. (2018). Baidu apollo em motion planner. *arXiv preprint arXiv:1807.08048*.

Fang, L., Jiang, Q., Shi, J., and Zhou, B. (2020). Tpnet: Trajectory proposal network for motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6797–6806.

Feng, L., Li, Q., Peng, Z., Tan, S., and Zhou, B. (2023). Trafficgen: Learning to generate diverse and realistic traffic scenarios. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3567–3575. IEEE.

Filos, A., Tigkas, P., McAllister, R., Rhinehart, N., Levine, S., and Gal, Y. (2020). Can autonomous vehicles identify, recover from, and adapt to distribution shifts? In *International Conference on Machine Learning*, pages 3145–3153. PMLR.

Fisac, J. F., Bronstein, E., Stefansson, E., Sadigh, D., Sastry, S. S., and Dragan, A. D. (2019). Hierarchical game-theoretic planning for autonomous vehicles. In *2019 International conference on robotics and automation (ICRA)*, pages 9590–9596. IEEE.

Fu, D., Li, X., Wen, L., Dou, M., Cai, P., Shi, B., and Qiao, Y. (2024). Drive like a human: Rethinking autonomous driving with large language models. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 910–919.

Galceran, E., Cunningham, A. G., Eustice, R. M., and Olson, E. (2017). Multipolicy decision-making for autonomous driving via changepoint-based behavior prediction: Theory and experiment. *Autonomous Robots*, **41**, 1367–1382.

Gambi, A., Nguyen, V., Ahmed, J., and Fraser, G. (2022). Generating critical driving scenarios from accident sketches. In *2022 IEEE International Conference On Artificial Intelligence Testing (AITest)*, pages 95–102. IEEE.

Gao, J., Sun, C., Zhao, H., Shen, Y., Anguelov, D., Li, C., and Schmid, C. (2020). Vectornet: Encoding hd maps and agent dynamics from vectorized representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11525–11533.

Geiger, P. and Straehle, C.-N. (2021). Learning game-theoretic models of multiagent trajectories using implicit layers. In *Proceedings of AAAI CAI*, volume 35, pages 4950–4958.

Gilles, T., Sabatini, S., Tsishkou, D., Stanciulescu, B., and Moutarde, F. (2021a). Home: Heatmap output for future motion estimation. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 500–507. IEEE.

Gilles, T., Sabatini, S., Tsishkou, D., Stanciulescu, B., and Moutarde, F. (2021b). Thomas: Trajectory heatmap output with learned multi-agent sampling. *arXiv preprint arXiv:2110.06607*.

Gilles, T., Sabatini, S., Tsishkou, D., Stanciulescu, B., and Moutarde, F. (2022). Gohome: Graph-oriented heatmap output for future motion estimation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 9107–9114. IEEE.

Girgis, R., Golemo, F., Codevilla, F., Weiss, M., D'Souza, J. A., Kahou, S. E., Heide, F., and Pal, C. (2021). Latent variable sequential set transformers for joint multi-agent motion prediction. *arXiv preprint arXiv:2104.00563*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2020). Generative adversarial networks. *Communications of the ACM*.

Greer, R., Deo, N., and Trivedi, M. (2021). Trajectory prediction in autonomous driving with a lane heading auxiliary loss. *RA-L*, **6**(3), 4907–4914.

Grigorescu, S., Trasnea, B., Cocias, T., and Macesanu, G. (2020). A survey of deep learning techniques for autonomous driving. *Journal of Field Robotics*, **37**(3), 362–386.

Gruyer, D., Magnier, V., Hamdi, K., Claussmann, L., Orfila, O., and Rakotonirainy, A. (2017). Perception, information processing and modeling: Critical stages for autonomous driving applications. *Annual Reviews in Control*, **44**, 323–341.

Gu, J., Sun, C., and Zhao, H. (2021). Densetnt: End-to-end trajectory prediction from dense goal sets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15303–15312.

Gulino, C., Fu, J., Luo, W., Tucker, G., Bronstein, E., Lu, Y., Harb, J., Pan, X., Wang, Y., Chen, X., *et al.* (2024). Waymax: An accelerated, data-driven simulator for large-scale autonomous driving research. *Advances in NeurIPS*, **36**.

Guo, Z., Gao, X., Zhou, J., Cai, X., and Shi, B. (2023). Scenedm: Scene-level multi-agent trajectory generation with consistent diffusion models. *arXiv:2311.15736*.

Gupta, A., Johnson, J., Fei-Fei, L., Savarese, S., and Alahi, A. (2018). Social gan: Socially acceptable trajectories with generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Hagedorn, S., Hallgarten, M., Stoll, M., and Condurache, A. (2023). Rethinking integration of prediction and planning in deep learning-based automated driving systems: a review. *arXiv preprint arXiv:2308.05731*.

Hagedorn, S., Milich, M., and Condurache, A. P. (2024). Pioneering se (2)-equivariant trajectory planning for automated driving. *arXiv:2403.11304*.

Hallgarten, M., Stoll, M., and Zell, A. (2023a). From prediction to planning with goal conditioned lane graph traversals. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 951–958. IEEE.

Hallgarten, M., Kisa, I., Stoll, M., and Zell, A. (2023b). Stay on track: A frenet wrapper to overcome off-road trajectories in vehicle motion prediction. *arXiv preprint arXiv:2306.00605*.

Hallgarten, M., Zapata, J., Stoll, M., Renz, K., and Zell, A. (2024). Can vehicle motion planning generalize to realistic long-tail scenarios? *arXiv preprint arXiv:2404.07569*.

Hang, P., Lv, C., Huang, C., Xing, Y., and Hu, Z. (2021). Cooperative decision making of connected automated vehicles at multi-lane merging zone: A coalitional game approach. *IEEE Transactions on Intelligent Transportation Systems*, **23**(4), 3829–3841.

Hanselmann, N., Renz, K., Chitta, K., Bhattacharyya, A., and Geiger, A. (2022). King: Generating safety-critical driving scenarios for robust imitation via kinematics gradients. In *European Conference on Computer Vision*, pages 335–352. Springer.

Hardy, J. and Campbell, M. (2013). Contingency planning over probabilistic obstacle predictions for autonomous road vehicles. *IEEE Transactions on Robotics*, **29**(4), 913–929.

Hauer, F., Gerostathopoulos, I., Schmidt, T., and Pretschner, A. (2020). Clustering traffic scenarios using mental models as little as possible. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 1007–1012. IEEE.

Hawke, J., Shen, R., Gurau, C., Sharma, S., Reda, D., Nikolov, N., Mazur, P., Micklethwaite, S., Griffiths, N., Shah, A., *et al.* (2020). Urban driving with conditional imitation learning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 251–257. IEEE.

Hazard, C., Bhagat, A., Buddharaju, B. R., Liu, Z., Shao, Y., Lu, L., Omari, S., and Cui, H. (2022). Importance is in your attention: agent importance prediction for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2532–2535.

Hecker, S., Dai, D., and Van Gool, L. (2018). End-to-end learning of driving models with surround-view cameras and route planners. In *Proceedings of the european conference on computer vision (eccv)*, pages 435–453.

Ho, J., Kalchbrenner, N., Weissenborn, D., and Salimans, T. (2019). Axial attention in multidimensional transformers. *arXiv preprint arXiv:1912.12180*.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, **9**(8), 1735–1780.

Hong, J., Sapp, B., and Philbin, J. (2019). Rules of the road: Predicting driving behavior with a convolutional model of semantic interactions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Houenou, A., Bonnifait, P., Cherfaoui, V., and Yao, W. (2013). Vehicle trajectory prediction based on motion model and maneuver recognition. In *2013 IEEE/RSJ international conference on intelligent robots and systems*, pages 4363–4369. IEEE.

Hu, H., Wang, Q., Zhang, Z., Li, Z., and Gao, Z. (2023a). Holistic transformer: A joint neural network for trajectory prediction and decision-making of autonomous vehicles. *Pattern Recognition*, **141**, 109592.

Hu, P., Huang, A., Dolan, J., Held, D., and Ramanan, D. (2021). Safe local motion planning with self-supervised freespace forecasting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12732–12741.

Hu, S., Chen, L., Wu, P., Li, H., Yan, J., and Tao, D. (2022). St-p3: End-to-end vision-based autonomous driving via spatial-temporal feature learning. In *European Conference on Computer Vision*, pages 533–549. Springer.

Hu, Y., Li, K., Liang, P., Qian, J., Yang, Z., Zhang, H., Shao, W., Ding, Z., Xu, W., and Liu, Q. (2023b). Imitation with spatial-temporal heatmap: 2nd place solution for nuplan challenge. *arXiv preprint arXiv:2306.15700*.

Hu, Y., Yang, J., Chen, L., Li, K., Sima, C., Zhu, X., Chai, S., Du, S., Lin, T., Wang, W., *et al.* (2023c). Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17853–17862.

Huang, Y. and Chen, Y. (2020). Survey of state-of-art autonomous driving technologies with deep learning. In *2020 IEEE 20th international conference on software quality, reliability and security companion (QRS-C)*, pages 221–228. IEEE.

Huang, Y., Du, J., Yang, Z., Zhou, Z., Zhang, L., and Chen, H. (2022a). A survey on trajectory-prediction methods for autonomous driving. *IEEE T-IV*, **7**(3), 652–674.

Huang, Z., Liu, H., Wu, J., and Lv, C. (2022b). Differentiable integrated motion prediction and planning with learnable cost function for autonomous driving. *arXiv preprint arXiv:2207.10422*.

Huang, Z., Karkus, P., Ivanovic, B., Chen, Y., Pavone, M., and Lv, C. (2023a). Dtpp: Differentiable joint conditional prediction and cost evaluation for tree policy planning in autonomous driving. *arXiv:2310.05885*.

Huang, Z., Liu, H., and Lv, C. (2023b). Gameformer: Game-theoretic modeling and learning of transformer-based interactive prediction and planning for autonomous driving. *arXiv preprint arXiv:2303.05760*.

Huang, Z., Liu, H., Mo, X., and Lyu, C. (2023c). Gameformer planner: A learning-enabled interactive prediction and planning framework for autonomous vehicles. *URL https://opendrivelab. com/e2ead/AD23Challenge/Track_4_AID. pdf*.

Hubmann, C., Schulz, J., Becker, M., Althoff, D., and Stiller, C. (2018). Automated driving in uncertain environments: Planning with interaction and uncertain maneuver prediction. *IEEE T-IV*, **3**(1), 5–17.

Igl, M., Kim, D., Kuefler, A., Mougin, P., Shah, P., Shiarlis, K., Anguelov, D., Palatucci, M., White, B., and Whiteson, S. (2022). Symphony: Learning realistic and diverse agents for autonomous driving simulation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2445–2451. IEEE.

Indaheng, F., Kim, E., Viswanadha, K., Shenoy, J., Kim, J., Fremont, D. J., and Seshia, S. A. (2021). A scenario-based platform for testing autonomous vehicle behavior prediction models in simulation. *arXiv preprint arXiv:2110.14870*.

Ivanovic, B. and Pavone, M. (2019). The trajectron: Probabilistic multi-agent trajectory modeling with dynamic spatiotemporal graphs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2375–2384.

Ivanovic, B., Leung, K., Schmerling, E., and Pavone, M. (2020). Multimodal deep generative models for trajectory prediction: A conditional variational autoencoder approach. *IEEE Robotics and Automation Letters*.

Jaeger, B., Chitta, K., and Geiger, A. (2023). Hidden biases of end-to-end driving models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8240–8249.

Janjoš, F., Dolgov, M., and Zöllner, J. M. (2021). Self-supervised action-space prediction for automated driving. In *2021 IEEE Intelligent Vehicles Symposium (IV)*, pages 200–207. IEEE.

Janjoš, F., Dolgov, M., Kurić, M., Shen, Y., and Zöllner, J. M. (2022a). San: Scene anchor networks for joint action-space prediction. In *2022 IEEE Intelligent Vehicles Symposium (IV)*, pages 1751–1756. IEEE.

Janjoš, F., Dolgov, M., Kurić, M., Shen, Y., and Zöllner, J. M. (2022b). San: Scene anchor networks for joint action-space prediction. In *2022 IEEE Intelligent Vehicles Symposium (IV)*. IEEE.

Janjoš, F., Dolgov, M., and Zöllner, J. M. (2022c). Starnet: Joint action-space prediction with star graphs and implicit global-frame self-attention. In *2022 IEEE Intelligent Vehicles Symposium (IV)*, pages 280–286. IEEE.

Janjoš, F., Keller, M., Dolgov, M., and Zöllner, J. M. (2023a). Bridging the gap between multi-step and one-shot trajectory prediction via self-supervision. In *2023 IEEE Intelligent Vehicles Symposium (IV)*. IEEE.

Janjoš, F., Hallgarten, M., Knittel, A., Dolgov, M., Zell, A., and Zöllner, J. M. (2023b). Conditional unscented autoencoders for trajectory prediction. *arXiv preprint arXiv:2310.19944*.

Janjoš, F., Rosenbaum, L., Dolgov, M., and Zöllner, J. M. (2023c). Unscented Autoencoder. In *40th International Conference on Machine Learning (ICML)*.

Jia, X., Sun, L., Zhao, H., Tomizuka, M., and Zhan, W. (2022). Multi-agent trajectory prediction by combining egocentric and allocentric views. In *Conference on Robot Learning*, pages 1434–1443. PMLR.

Jiang, B., Chen, S., Xu, Q., Liao, B., Chen, J., Zhou, H., Zhang, Q., Liu, W., Huang, C., and Wang, X. (2023a). Vad: Vectorized scene representation for efficient autonomous driving. In *IEEE/CVF ICCV*, pages 8340–8350.

Jiang, C., Cornman, A., Park, C., Sapp, B., Zhou, Y., Anguelov, D., *et al.* (2023b). Motiondiffuser: Controllable multi-agent motion prediction using diffusion. In *IEEE/CVF CVPR*, pages 9644–9653.

Jin, B., Liu, X., Zheng, Y., Li, P., Zhao, H., Zhang, T., Zheng, Y., Zhou, G., and Liu, J. (2023). Adapt: Action-aware driving caption transformer. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7554–7561. IEEE.

Julier, S., Uhlmann, J., and Durrant-Whyte, H. F. (2000). A new method for the nonlinear transformation of means and covariances in filters and estimators. *IEEE Transactions on automatic control*.

Julier, S. J. and Uhlmann, J. K. (2004). Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*.

Kamenev, A., Wang, L., Bohan, O. B., Kulkarni, I., Kartal, B., Molchanov, A., Birchfield, S., Nistér, D., and Smolyanskiy, N. (2022). Predictionnet: Real-time joint probabilistic traffic prediction for planning, control, and simulation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 8936–8942. IEEE.

Kant, K. and Zucker, S. W. (1986). Toward efficient trajectory planning: The path-velocity decomposition. *The international journal of robotics research*, **5**(3), 72–89.

Karkus, P., Ivanovic, B., Mannor, S., and Pavone, M. (2023). Diffstack: A differentiable and modular control stack for autonomous vehicles. In *Conference on Robot Learning*, pages 2170–2180. PMLR.

Kesting, A., Treiber, M., and Helbing, D. (2007). General lane-changing model mobil for car-following models. *Transportation Research Record*, **1999**(1), 86–94.

Kesting, A., Treiber, M., and Helbing, D. (2010). Enhanced intelligent driver model to access the impact of driving strategies on traffic capacity. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, **368**(1928), 4585–4605.

Khandelwal, S., Qi, W., Singh, J., Hartnett, A., and Ramanan, D. (2020). What-if motion prediction for autonomous driving. *arXiv preprint arXiv:2008.10587*.

Khayatkhoei, M., Singh, M. K., and Elgammal, A. (2018). Disconnected manifold learning for generative adversarial networks. *Advances in Neural Information Processing Systems*.

Kim, B., Kang, C. M., Kim, J., Lee, S. H., Chung, C. C., and Choi, J. W. (2017). Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 399–404. IEEE.

Kim, B., Park, S. H., Lee, S., Khoshimjonov, E., Kum, D., Kim, J., Kim, J. S., and Choi, J. W. (2021). Lapred: Lane-aware prediction of multi-modal future trajectories of dynamic agents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14636–14645.

Kim, J., Moon, S., Rohrbach, A., Darrell, T., and Canny, J. (2020). Advisable learning for self-driving vehicles by internalizing observation-to-action rules. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9661–9670.

Kim, J., Mahjourian, R., Ettinger, S., Bansal, M., White, B., Sapp, B., and Anguelov, D. (2022). Stopnet: Scalable trajectory and occupancy prediction for urban autonomous driving. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 8957–8963. IEEE.

Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Kipf, T. N. and Welling, M. (2016). Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Klimke, M., Völz, B., and Buchholz, M. (2022). Cooperative behavior planning for automated driving using graph neural networks. In *2022 IEEE Intelligent Vehicles Symposium (IV)*, pages 167–174. IEEE.

Klimke, M., Völz, B., and Buchholz, M. (2023). Automatic intersection management in mixed traffic using reinforcement learning and graph neural networks. In *2023 IEEE Intelligent Vehicles Symposium (IV)*, pages 1–8. IEEE.

Klischat, M. and Althoff, M. (2019). Generating critical test scenarios for automated vehicles with evolutionary algorithms. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 2352–2358. IEEE.

Klück, F., Li, Y., Tao, J., and Wotawa, F. (2023). An empirical comparison of combinatorial testing and search-based testing in the context of automated and autonomous driving systems. *Information and Software Technology*, **160**, 107225.

Knittel, A., Hawasly, M., Albrecht, S. V., Redford, J., and Ramamoorthy, S. (2022). Dipa: Diverse and probabilistically accurate interactive prediction. *arXiv preprint arXiv:2210.06106*.

Knittel, A., Hawasly, M., Albrecht, S. V., Redford, J., and Ramamoorthy, S. (2023). Dipa: Probabilistic multi-modal interactive prediction for autonomous driving. *IEEE Robotics and Automation Letters*.

Konev, S. (2022). Mpa: Multipath++ based architecture for motion prediction. *arXiv preprint arXiv:2206.10041*.

Konev, S., Brodt, K., and Sanakoyeu, A. (2022). Motioncnn: a strong baseline for motion prediction in autonomous driving. *arXiv preprint arXiv:2206.02163*.

Kong, Z., Guo, J., Li, A., and Liu, C. (2020). Physgan: Generating physical-world-resilient adversarial examples for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14254–14263.

Kosaraju, V., Sadeghian, A., Martín-Martín, R., Reid, I., Rezatofighi, H., and Savarese, S. (2019). Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks. *Advances in Neural Information Processing Systems*.

Kothari, P., Kreiss, S., and Alahi, A. (2021). Human trajectory forecasting in crowds: A deep learning perspective. *IEEE Transactions on Intelligent Transportation Systems*, **23**(7), 7386–7400.

Krajewski, R., Bock, J., Kloeker, L., and Eckstein, L. (2018). The highd dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems. In *2018 21st international conference on intelligent transportation systems (ITSC)*, pages 2118–2125. IEEE.

Krajewski, R., Moers, T., Bock, J., Vater, L., and Eckstein, L. (2020). The round dataset: A drone dataset of road user trajectories at roundabouts in germany. In *2020 IEEE*

23rd International Conference on Intelligent Transportation Systems (ITSC), pages 1–6. IEEE.

Kumar, E., Zhang, Y., Pini, S., Stent, S., Ferreira, A., Zagoruyko, S., and Perone, C. S. (2022). Cw-erm: Improving autonomous driving planning with closed-loop weighted empirical risk minimization. *arXiv preprint arXiv:2210.02174*.

Kuutti, S., Bowden, R., Jin, Y., Barber, P., and Fallah, S. (2020). A survey of deep learning applications to autonomous vehicle control. *IEEE Transactions on Intelligent Transportation Systems*, **22**(2), 712–733.

Langner, J., Bach, J., Ries, L., Otten, S., Holzäpfel, M., and Sax, E. (2018). Estimating the uniqueness of test scenarios derived from recorded real-world-driving-data using autoencoders. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1860–1866. IEEE.

LeCun, Y., Boser, B., Denker, J., Henderson, D., Hubbard, W., and Jackel, L. (1989). Handwritten digit recognition with a back-propagation network. *Advances in neural information processing systems*, **2**.

Lee, M., Sohn, S. S., Moon, S., Yoon, S., Kapadia, M., and Pavlovic, V. (2022). Muse-vae: multi-scale vae for environment-aware long term trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Lee, N., Choi, W., Vernaza, P., Choy, C. B., Torr, P. H., and Chandraker, M. (2017). Desire: Distant future prediction in dynamic scenes with interacting agents. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 336–345.

Lefèvre, S., Vasquez, D., and Laugier, C. (2014). A survey on motion prediction and risk assessment for intelligent vehicles. *ROBOMECH journal*, **1**(1), 1–14.

Leon, F. and Gavrilescu, M. (2021). A review of tracking and trajectory prediction methods for autonomous driving. *Mathematics*, **9**(6), 660.

Leonard, J., How, J., Teller, S., Berger, M., Campbell, S., Fiore, G., Fletcher, L., Frazzoli, E., Huang, A., Karaman, S., *et al.* (2008). A perception-driven autonomous urban vehicle. *Journal of Field Robotics*, **25**(10), 727–774.

Leurent, E. (2018). An environment for autonomous driving decision-making. https://github.com/eleurent/highway-env.

Li, D., Zhang, Q., Xia, Z., Zhang, K., Yi, M., Jin, W., and Zhao, D. (2023a). Planning-inspired hierarchical trajectory prediction for autonomous driving. *arXiv preprint arXiv:2304.11295*.

Li, L. L., Yang, B., Liang, M., Zeng, W., Ren, M., Segal, S., and Urtasun, R. (2020). End-to-end contextual perception and prediction with interaction transformer. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5784–5791. IEEE.

Li, N., Oyler, D. W., Zhang, M., Yildiz, Y., Kolmanovsky, I., and Girard, A. R. (2017). Game theoretic modeling of driver and vehicle interactions for verification and validation of autonomous vehicle control systems. *IEEE Transactions on control systems technology*, **26**(5), 1782–1797.

Li, P., Pei, X., Chen, Z., Zhou, X., and Xu, J. (2022a). Human-like motion planning of autonomous vehicle based on probabilistic trajectory prediction. *Applied Soft Computing*, **118**, 108499.

Li, T., Zhang, L., Liu, S., and Shen, S. (2023b). Marc: Multipolicy and risk-aware contingency planning for autonomous driving. *IEEE Robotics and Automation Letters*.

Li, T., Zhang, L., Liu, S., and Shen, S. (2024a). Multi-modal integrated prediction and decision-making with adaptive interaction modality explorations. *arXiv:2408.13742*.

Li, X., Ying, X., and Chuah, M. C. (2019a). Grip++: Enhanced graph-based interaction-aware trajectory prediction for autonomous driving. *arXiv preprint arXiv:1907.07792*.

Li, X., Ying, X., and Chuah, M. C. (2019b). Grip: Graph-based interaction-aware trajectory prediction. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 3960–3966. IEEE.

Li, Z., Motoyoshi, T., Sasaki, K., Ogata, T., and Sugano, S. (2018). Rethinking self-driving: Multi-task knowledge for better generalization and accident explanation ability. *arXiv preprint arXiv:1809.11100*.

Li, Z., Wang, W., Li, H., Xie, E., Sima, C., Lu, T., Qiao, Y., and Dai, J. (2022b). Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In *ECCV*, pages 1–18. Springer.

Li, Z., Li, K., Wang, S., Lan, S., Yu, Z., Ji, Y., Li, Z., Zhu, Z., Kautz, J., Wu, Z., *et al.* (2024b). Hydra-mdp: End-to-end multimodal planning with multi-target hydra-distillation. *arXiv:2406.06978*.

Li, Z., Yu, Z., Lan, S., Li, J., Kautz, J., Lu, T., and Alvarez, J. M. (2024c). Is ego status all you need for open-loop end-to-end autonomous driving? In *CVPR*, pages 14864–14873.

Liang, M., Yang, B., Hu, R., Chen, Y., Liao, R., Feng, S., and Urtasun, R. (2020). Learning lane graph representations for motion forecasting. In *Computer Vision–ECCV 2020:*

16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16, pages 541–556. Springer.

Liao, H., Li, Z., Shen, H., Zeng, W., Liao, D., Li, G., and Xu, C. (2024). Bat: Behavior-aware human-like trajectory prediction for autonomous driving. In *AAAI CAI*, volume 38, pages 10332–10340.

Liu, C., Lee, S., Varnhagen, S., and Tseng, H. E. (2017). Path planning for autonomous vehicles using model predictive control. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 174–179. IEEE.

Liu, J., Zeng, W., Urtasun, R., and Yumer, E. (2021a). Deep structured reactive planning. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4897–4904. IEEE.

Liu, J., Mao, X., Fang, Y., Zhu, D., and Meng, M. Q.-H. (2021b). A survey on deep-learning approaches for vehicle trajectory prediction in autonomous driving. In *2021 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 978–985. IEEE.

Liu, M., Cheng, H., Chen, L., Broszio, H., Li, J., Zhao, R., Sester, M., and Yang, M. Y. (2024). Laformer: Trajectory prediction for autonomous driving with lane-aware scene constraints. In *IEEE/CVF CVPR*, pages 2039–2049.

Liu, Y., Zhang, J., Fang, L., Jiang, Q., and Zhou, B. (2021c). Multimodal motion prediction with stacked transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7577–7586.

Lu, Q., Han, W., Ling, J., Wang, M., Chen, H., Varadarajan, B., and Covington, P. (2022). Kemp: Keyframe-based hierarchical end-to-end deep model for long-term trajectory prediction. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 646–652. IEEE.

Luo, C., Sun, L., Dabiri, D., and Yuille, A. (2020). Probabilistic multi-modal trajectory prediction with lane attention for autonomous vehicles. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2370–2376. IEEE.

Luo, W., Yang, B., and Urtasun, R. (2018). Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 3569–3577.

Luong, M.-T., Pham, H., and Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Mahjourian, R., Kim, J., Chai, Y., Tan, M., Sapp, B., and Anguelov, D. (2022). Occupancy flow fields for motion forecasting in autonomous driving. *IEEE Robotics and Automation Letters*, **7**(2), 5639–5646.

Mao, J., Qian, Y., Zhao, H., and Wang, Y. (2023a). Gpt-driver: Learning to drive with gpt. *arXiv preprint arXiv:2310.01415*.

Mao, J., Ye, J., Qian, Y., Pavone, M., and Wang, Y. (2023b). A language agent for autonomous driving. *arXiv preprint arXiv:2311.10813*.

McDuff, D., Song, Y., Lee, J., Vineet, V., Vemprala, S., Gyde, N. A., Salman, H., Ma, S., Sohn, K., and Kapoor, A. (2022). Causalcity: Complex simulations with agency for causal discovery and reasoning. In *Conference on Causal Learning and Reasoning*, pages 559–575. PMLR.

Menzel, T., Bagschik, G., and Maurer, M. (2018). Scenarios for development, test and validation of automated vehicles. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1821–1827. IEEE.

Mercat, J., Gilles, T., El Zoghby, N., Sandou, G., Beauvois, D., and Gil, G. P. (2020). Multi-head attention for multi-modal joint vehicle motion forecasting. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9638–9644. IEEE.

Messaoud, K., Deo, N., Trivedi, M. M., and Nashashibi, F. (2020). Multi-head attention with joint agent-map representation for trajectory prediction in autonomous driving. *arXiv:2005.02545*, **2**.

Mészáros, A., Alonso-Mora, J., and Kober, J. (2023). Trajflow: Learning the distribution over trajectories. *arXiv preprint arXiv:2304.05166*.

Mi, L., Zhao, H., Nash, C., Jin, X., Gao, J., Sun, C., Schmid, C., Shavit, N., Chai, Y., and Anguelov, D. (2021). Hdmapgen: A hierarchical graph generative model of high definition maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4227–4236.

Mo, X., Huang, Z., Xing, Y., and Lv, C. (2022). Multi-agent trajectory prediction with heterogeneous edge-enhanced graph attention network. *IEEE Transactions on Intelligent Transportation Systems*, **23**(7), 9554–9567.

Moers, T., Vater, L., Krajewski, R., Bock, J., Zlocki, A., and Eckstein, L. (2022). The exid dataset: A real-world trajectory dataset of highly interactive highway scenarios in germany. In *2022 IEEE Intelligent Vehicles Symposium (IV)*, pages 958–964.

Montremerlo, M., Beeker, J., Bhat, S., and Dahlkamp, H. (2008). The stanford entry in the urban challenge. *Journal of Field Robotics*, **7**(9), 468–492.

Moravec, H. (1988). *Mind children: The future of robot and human intelligence*. Harvard University Press.

Mozaffari, S., Al-Jarrah, O. Y., Dianati, M., Jennings, P., and Mouzakitis, A. (2020). Deep learning-based vehicle behavior prediction for autonomous driving applications: A review. *IEEE Transactions on Intelligent Transportation Systems*, **23**(1), 33–47.

Mu, N., Ji, J., Yang, Z., Harada, N., Tang, H., Chen, K., Qi, C. R., Ge, R., Goel, K., Yang, Z., *et al.* (2024). Most: Multi-modality scene tokenization for motion prediction. In *CVPR*, pages 14988–14999.

Müller, M., Dosovitskiy, A., Ghanem, B., and Koltun, V. (2018). Driving policy transfer via modularity and abstraction. *arXiv preprint arXiv:1804.09364*.

Muller, U., Ben, J., Cosatto, E., Flepp, B., and Cun, Y. (2005). Off-road obstacle avoidance through end-to-end learning. *Advances in neural information processing systems*, **18**.

Narayanan, S., Moslemi, R., Pittaluga, F., Liu, B., and Chandraker, M. (2021). Divide-and-conquer for lane-aware diverse trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15799–15808.

Nayakanti, N., Al-Rfou, R., Zhou, A., Goel, K., Refaat, K. S., and Sapp, B. (2022). Wayformer: Motion forecasting via simple & efficient attention networks. *arXiv preprint arXiv:2207.05844*.

Ngiam, J., Caine, B., Han, W., Yang, B., Chai, Y., Sun, P., Zhou, Y., Yi, X., Alsharif, O., Nguyen, P., *et al.* (2019). Starnet: Targeted computation for object detection in point clouds. *arXiv preprint arXiv:1908.11069*.

Ngiam, J., Vasudevan, V., Caine, B., Zhang, Z., Chiang, H.-T. L., Ling, J., Roelofs, R., Bewley, A., Liu, C., Venugopal, A., *et al.* (2021). Scene transformer: A unified architecture for predicting future trajectories of multiple agents. In *International Conference on Learning Representations*.

Ngiam, J., Vasudevan, V., Caine, B., Zhang, Z., Chiang, H.-T. L., Ling, J., Roelofs, R., Bewley, A., Liu, C., Venugopal, A., *et al.* (2022). Scene transformer: A unified architecture for predicting future trajectories of multiple agents. In *International Conference on Learning Representations*.

Niedoba, M., Cui, H., Luo, K., Hegde, D., Chou, F.-C., and Djuric, N. (2019). Improving movement prediction of traffic actors using off-road loss and bias mitigation. In *Workshop on'Machine Learning for Autonomous Driving'at Conference on Neural Information Processing Systems*, volume 1, page 2.

Ohn-Bar, E., Prakash, A., Behl, A., Chitta, K., and Geiger, A. (2020). Learning situational driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11296–11305.

Pan, J., Sun, H., Xu, K., Jiang, Y., Xiao, X., Hu, J., and Miao, J. (2020). Lane-attention: Predicting vehicles' moving trajectories by learning their attention over lanes. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7949–7956. IEEE.

Park, S. H., Lee, G., Seo, J., Bhat, M., Kang, M., Francis, J., Jadhav, A., Liang, P. P., and Morency, L.-P. (2020). Diverse and admissible trajectory forecasting through multimodal context understanding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pages 282–298. Springer.

Phan-Minh, T., Grigore, E. C., Boulton, F. A., Beijbom, O., and Wolff, E. M. (2020). Covernet: Multimodal behavior prediction using trajectory sets. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14074–14083.

Phan-Minh, T., Howington, F., Chu, T.-S., Tomov, M. S., Beaudoin, R. E., Lee, S. U., Li, N., Dicle, C., Findler, S., Suarez-Ruiz, F., *et al.* (2023). Driveirl: Drive in real life with inverse reinforcement learning. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1544–1550. IEEE.

Philion, J. and Fidler, S. (2020). Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *ECCV*, pages 194–210. Springer.

Philion, J., Peng, X. B., and Fidler, S. (2024). Trajeglish: Traffic modeling as next-token prediction. In *ICLR*.

Pini, S., Perone, C. S., Ahuja, A., Ferreira, A. S. R., Niendorf, M., and Zagoruyko, S. (2022). Safe real-world autonomous driving by learning to predict and plan with a mixture of experts. *arXiv preprint arXiv:2211.02131*.

Pittner, M., Janai, J., and Condurache, A. P. (2024). Lanecpp: Continuous 3d lane detection using physical priors. In *IEEE/CVF CVPR*, pages 10639–10648.

Poddar, S., Mavrogiannis, C., and Srinivasa, S. S. (2023). From crowd motion prediction to robot navigation in crowds. *arXiv preprint arXiv:2303.01424*.

Polack, P., Altché, F., d'Andréa Novel, B., and de La Fortelle, A. (2017). The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles? In *2017 IEEE intelligent vehicles symposium (IV)*, pages 812–818. IEEE.

Pomerleau, D. A. (1988). Alvinn: An autonomous land vehicle in a neural network. *Advances in neural information processing systems*, **1**.

Postnikov, A., Gamayunov, A., and Ferrer, G. (2021). Transformer based trajectory prediction. *arXiv preprint arXiv:2112.04350*.

Prakash, A., Behl, A., Ohn-Bar, E., Chitta, K., and Geiger, A. (2020). Exploring data aggregation in policy learning for vision-based urban autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11763–11773.

Prakash, A., Chitta, K., and Geiger, A. (2021). Multi-modal fusion transformer for end-to-end autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7077–7087.

Pulver, H., Eiras, F., Carozza, L., Hawasly, M., Albrecht, S. V., and Ramamoorthy, S. (2021). Pilot: Efficient planning by imitation learning and optimisation for safe autonomous driving. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1442–1449. IEEE.

Qi, C. R., Zhou, Y., Najibi, M., Sun, P., Vo, K., Deng, B., and Anguelov, D. (2021). Offboard 3d object detection from point cloud sequences. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6134–6144.

Quintanar, A., Fernández-Llorca, D., Parra, I., Izquierdo, R., and Sotelo, M. (2021). Predicting vehicles trajectories in urban scenarios with transformer networks and augmented information. In *2021 IEEE Intelligent Vehicles Symposium (IV)*, pages 1051–1056. IEEE.

Rajamani, R. (2011). *Vehicle dynamics and control*. Springer Science & Business Media.

Reiter, R., Quirynen, R., Diehl, M., and Di Cairano, S. (2024). Equivariant deep learning of mixed-integer optimal control solutions for vehicle decision making and motion planning. *IEEE TCST*.

Rempe, D., Philion, J., Guibas, L. J., Fidler, S., and Litany, O. (2022). Generating useful accident-prone driving scenarios via a learned traffic prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17305–17315.

Renz, K., Chitta, K., Mercea, O.-B., Koepke, A., Akata, Z., and Geiger, A. (2022). Plant: Explainable planning transformers via object-level representations. *arXiv preprint arXiv:2210.14222*.

Rezende, D. and Mohamed, S. (2015). Variational inference with normalizing flows. In *ICML*, pages 1530–1538. PMLR.

Rhinehart, N., McAllister, R., and Levine, S. (2018a). Deep imitative models for flexible inference, planning, and control. *arXiv preprint arXiv:1810.06544*.

Rhinehart, N., Kitani, K. M., and Vernaza, P. (2018b). R2p2: A reparameterized push-forward policy for diverse, precise generative path forecasting. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 772–788.

Rhinehart, N., McAllister, R., Kitani, K., and Levine, S. (2019). Precog: Prediction conditioned on goals in visual multi-agent settings. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2821–2830.

Rhinehart, N., He, J., Packer, C., Wright, M. A., McAllister, R., Gonzalez, J. E., and Levine, S. (2021). Contingencies from observations: Tractable contingency planning with learned behavior models. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13663–13669. IEEE.

Ridel, D., Deo, N., Wolf, D., and Trivedi, M. (2020). Scene compliant trajectory forecast with agent-centric spatio-temporal grids. *IEEE RA-L*, **5**(2), 2816–2823.

Rocklage, E., Kraft, H., Karatas, A., and Seewig, J. (2017). Automated scenario generation for regression testing of autonomous vehicles. In *2017 ieee 20th international conference on intelligent transportation systems (itsc)*, pages 476–483. IEEE.

Rolfe, J. T. (2016). Discrete variational autoencoders. *arXiv preprint arXiv:1609.02200*.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, **65**(6), 386.

Ross, S., Gordon, G., and Bagnell, D. (2011). A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, **323**(6088), 533–536.

Saadatnejad, S., Li, S., Mordan, T., and Alahi, A. (2021). A shared representation for photorealistic driving simulators. *IEEE Transactions on Intelligent Transportation Systems*, **23**(8), 13835–13845.

Sadat, A., Ren, M., Pokrovsky, A., Lin, Y.-C., Yumer, E., and Urtasun, R. (2019). Jointly learnable behavior and trajectory planning for self-driving vehicles. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3949–3956. IEEE.

Sadat, A., Casas, S., Ren, M., Wu, X., Dhawan, P., and Urtasun, R. (2020). Perceive, predict, and plan: Safe motion planning through interpretable semantic representations. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIII 16*, pages 414–430. Springer.

Sadeghian, A., Kosaraju, V., Sadeghian, A., Hirose, N., Rezatofighi, H., and Savarese, S. (2019). Sophie: An attentive gan for predicting paths compliant to social and physical constraints. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Rrecognition*.

Sadigh, D., Sastry, S., Seshia, S. A., and Dragan, A. D. (2016). Planning for autonomous cars that leverage effects on human actions. In *Robotics: Science and systems*, volume 2, pages 1–9. Ann Arbor, MI, USA.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved techniques for training gans. *Advances in neural information processing systems*.

Salzmann, T., Ivanovic, B., Chakravarty, P., and Pavone, M. (2020). Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16*, pages 683–700. Springer.

Sato, T., Shen, J., Wang, N., Jia, Y., Lin, X., and Chen, Q. A. (2021). Dirty road can attack: Security of deep learning based automated lane centering under {Physical-World} attack. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 3309–3326.

Sauer, A., Savinov, N., and Geiger, A. (2018). Conditional affordance learning for driving in urban environments. In *Conference on robot learning*, pages 237–252. PMLR.

Schaal, S. (1996). Learning from demonstration. *Advances in neural information processing systems*, **9**.

Scheel, O., Bergamini, L., Wolczyk, M., Osiński, B., and Ondruska, P. (2022). Urban driver: Learning to drive from real-world demonstrations using policy gradients. In *Conference on Robot Learning*, pages 718–728. PMLR.

Schmerling, E., Leung, K., Vollprecht, W., and Pavone, M. (2018). Multimodal probabilistic model-based planning for human-robot interaction. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3399–3406. IEEE.

Schmidt, J., Jordan, J., Gritschneder, F., and Dietmayer, K. (2022). Crat-pred: Vehicle trajectory prediction with crystal graph convolutional neural networks and multi-head self-attention. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 7799–7805. IEEE.

Schöller, C. and Knoll, A. (2021). Flomo: Tractable motion prediction with normalizing flows. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE.

Schöller, C., Aravantinos, V., Lay, F., and Knoll, A. (2020). What the constant velocity model can teach us about pedestrian motion prediction. *IEEE Robotics and Automation Letters*, **5**(2), 1696–1703.

Schwarting, W., Alonso-Mora, J., and Rus, D. (2018). Planning and decision-making for autonomous vehicles. *Annual Review of Control, Robotics, and Autonomous Systems*, **1**, 187–210.

Ścibior, A., Lioutas, V., Reda, D., Bateni, P., and Wood, F. (2021). Imagining the road ahead: Multi-agent trajectory prediction via differentiable simulation. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. IEEE.

Sha, H., Mu, Y., Jiang, Y., Chen, L., Xu, C., Luo, P., Li, S. E., Tomizuka, M., Zhan, W., and Ding, M. (2023). Languagempc: Large language models as decision makers for autonomous driving. *arXiv preprint arXiv:2310.03026*.

Shalev-Shwartz, S., Shammah, S., and Shashua, A. (2017). On a formal model of safe and scalable self-driving cars. *arXiv preprint arXiv:1708.06374*.

Shao, H., Wang, L., Chen, R., Waslander, S. L., Li, H., and Liu, Y. (2023a). Reasonnet: End-to-end driving with temporal and global reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13723–13733.

Shao, H., Wang, L., Chen, R., Li, H., and Liu, Y. (2023b). Safety-enhanced autonomous driving using interpretable sensor fusion transformer. In *Conference on Robot Learning*, pages 726–737. PMLR.

Sharath, M. and Velaga, N. R. (2020). Enhanced intelligent driver model for two-dimensional motion planning in mixed traffic. *Transportation Research Part C: Emerging Technologies*, **120**, 102780.

Sheng, Z., Liu, L., Xue, S., Zhao, D., Jiang, M., and Li, D. (2022). A cooperation-aware lane change method for automated vehicles. *IEEE Transactions on Intelligent Transportation Systems*, **24**(3), 3236–3251.

Shi, S., Jiang, L., Dai, D., and Schiele, B. (2022a). Motion transformer with global intention localization and local movement refinement. *Advances in Neural Information Processing Systems*, **35**, 6531–6543.

Shi, S., Jiang, L., Dai, D., and Schiele, B. (2022b). Mtr-a: 1st place solution for 2022 waymo open dataset challenge–motion prediction. *arXiv preprint arXiv:2209.10033*.

Sima, C., Renz, K., Chitta, K., Chen, L., Zhang, H., Xie, C., Luo, P., Geiger, A., and Li, H. (2023). Drivelm: Driving with graph visual question answering. *arXiv preprint arXiv:2312.14150*.

Singh, D. and Srivastava, R. (2022). Multi-scale graph-transformer network for trajectory prediction of the autonomous vehicles. *Intelligent Service Robotics*, **15**(3), 307–320.

Sohn, K., Lee, H., and Yan, X. (2015). Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*.

Song, H., Ding, W., Chen, Y., Shen, S., Wang, M. Y., and Chen, Q. (2020). Pip: Planning-informed trajectory prediction for autonomous driving. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXI 16*, pages 598–614. Springer.

Song, H., Luan, D., Ding, W., Wang, M. Y., and Chen, Q. (2022). Learning to predict vehicle trajectories with model-based planning. In *Conference on Robot Learning*, pages 1035–1045. PMLR.

Sriram, N., Liu, B., Pittaluga, F., and Chandraker, M. (2020). Smart: Simultaneous multi-agent recurrent trajectory prediction. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVII 16*, pages 463–479. Springer.

Stoll, M., Mazzola, M., Dolgov, M., Mathes, J., and Möser, N. (2023). Scaling planning for automated driving using simplistic synthetic data. *arXiv preprint arXiv:2305.18942*.

Sun, Q., Huang, X., Gu, J., Williams, B. C., and Zhao, H. (2022). M2i: From factored marginal trajectory prediction to interactive prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6543–6552.

Sun, Q., Zhang, S., Ma, D., Shi, J., Li, D., Luo, S., Wang, Y., Xu, N., Cao, G., and Zhao, H. (2023). Large trajectory models are scalable motion predictors and planners. *arXiv preprint arXiv:2310.19620*.

Suo, S., Regalado, S., Casas, S., and Urtasun, R. (2021). Trafficsim: Learning to simulate realistic multi-agent behaviors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10400–10409.

Suo, S., Wong, K., Xu, J., Tu, J., Cui, A., Casas, S., and Urtasun, R. (2023). Mixsim: A hierarchical framework for mixed reality traffic simulation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Tang, C. and Salakhutdinov, R. R. (2019). Multiple futures prediction. *Advances in neural information processing systems*, **32**.

Tanielian, U., Issenhuth, T., Dohmatob, E., and Mary, J. (2020). Learning disconnected manifolds: a no gan's land. In *International Conference on Machine Learning*. PMLR.

Tassa, Y., Mansard, N., and Todorov, E. (2014). Control-limited differential dynamic programming. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1168–1175. IEEE.

Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G., *et al.* (2006). Stanley: The robot that won the darpa grand challenge. *Journal of field Robotics*, **23**(9), 661–692.

Tolstaya, E., Mahjourian, R., Downey, C., Vadarajan, B., Sapp, B., and Anguelov, D. (2021). Identifying driver interactions via conditional behavior prediction. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3473–3479. IEEE.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., *et al.* (2023). Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Treiber, M., Hennecke, A., and Helbing, D. (2000). Congested traffic states in empirical observations and microscopic simulations. *Physical review E*, **62**(2), 1805.

Truong, N. H., Mai, H. T., Tran, T. A., Tran, M. Q., Nguyen, D. D., and Pham, N. V. P. (2023). Paas: Planning as a service for reactive driving in carla leaderboard. In *2023 International Conference on System Science and Engineering (ICSSE)*, pages 101–107. IEEE.

Urmson, C., Anhalt, J., Bagnell, D., Baker, C., Bittner, R., Clark, M., Dolan, J., Duggins, D., Galatali, T., Geyer, C., *et al.* (2008). Autonomous driving in urban environments: Boss and the urban challenge. *Journal of field Robotics*, **25**(8), 425–466.

Varadarajan, B., Hefny, A., Srivastava, A., Refaat, K. S., Nayakanti, N., Cornman, A., Chen, K., Douillard, B., Lam, C. P., Anguelov, D., *et al.* (2022). Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 7814–7821. IEEE.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, **30**.

Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y., *et al.* (2017). Graph attention networks. *stat*, **1050**(20), 10–48550.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, **17**, 261–272.

Vitelli, M., Chang, Y., Ye, Y., Ferreira, A., Wołczyk, M., Osiński, B., Niendorf, M., Grimmett, H., Huang, Q., Jain, A., *et al.* (2022). Safetynet: Safe planning for real-world self-driving vehicles using machine-learned policies. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 897–904. IEEE.

Wachi, A. (2019). Failure-scenario maker for rule-based agent using multi-agent adversarial reinforcement learning and its application to autonomous driving. *arXiv preprint arXiv:1903.10654*.

Wang, D., Devin, C., Cai, Q.-Z., Yu, F., and Darrell, T. (2019a). Deep object-centric policies for autonomous driving. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8853–8859. IEEE.

Wang, D., Devin, C., Cai, Q.-Z., Krähenbühl, P., and Darrell, T. (2019b). Monocular plan view networks for autonomous driving. In *IROS*, pages 2876–2883. IEEE/RSJ.

Wang, J., Pun, A., Tu, J., Manivasagam, S., Sadat, A., Casas, S., Ren, M., and Urtasun, R. (2021). Advsim: Generating safety-critical scenarios for self-driving vehicles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9909–9918.

Wang, J., Ye, T., Gu, Z., and Chen, J. (2022a). Ltp: Lane-based trajectory prediction for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17134–17142.

Wang, W., Wang, L., Zhang, C., Liu, C., Sun, L., *et al.* (2022b). Social interactions for autonomous driving: A review and perspectives. *Foundations and Trends® in Robotics*, **10**(3-4), 198–376.

Wang, W., Xie, J., Hu, C., Zou, H., Fan, J., Tong, W., Wen, Y., Wu, S., Deng, H., Li, Z., *et al.* (2023a). Drivemlm: Aligning multi-modal large language models with behavioral planning states for autonomous driving. *arXiv preprint arXiv:2312.09245*.

Wang, X., Su, T., Da, F., and Yang, X. (2023b). Prophnet: Efficient agent-centric motion forecasting with anchor-informed proposals. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21995–22003.

Wang, Y., Sun, Y., Liu, Z., Sarma, S. E., Bronstein, M. M., and Solomon, J. M. (2019c). Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, **38**(5), 1–12.

Wang, Y., Zhou, H., Zhang, Z., Feng, C., Lin, H., Gao, C., Tang, Y., Zhao, Z., Zhang, S., Guo, J., *et al.* (2022c). Tenet: Transformer encoding network for effective temporal flow on motion prediction. *arXiv preprint arXiv:2207.00170*.

Wang, Y., Jiao, R., Lang, C., Zhan, S. S., Huang, C., Wang, Z., Yang, Z., and Zhu, Q. (2023c). Empowering autonomous driving with large language models: A safety perspective. *arXiv preprint arXiv:2312.00812*.

Wei, B., Ren, M., Zeng, W., Liang, M., Yang, B., and Urtasun, R. (2021). Perceive, attend, and drive: Learning spatial attention for safe self-driving. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4875–4881. IEEE.

Wen, L.-H. and Jo, K.-H. (2022). Deep learning-based perception systems for autonomous driving: A comprehensive survey. *Neurocomputing*.

Werling, M., Ziegler, J., Kammel, S., and Thrun, S. (2010). Optimal trajectory generation for dynamic street scenarios in a frenet frame. In *2010 IEEE international conference on robotics and automation*, pages 987–993. IEEE.

Wetmore, J. (2003). Driving the dream. the history and motivations behind 60 years of automated highway systems in america. *Automotive History Review*, **7**, 4–19.

Wilson, B., Qi, W., Agarwal, T., Lambert, J., Singh, J., Khandelwal, S., Pan, B., Kumar, R., Hartnett, A., Pontes, J. K., *et al.* (2023). Argoverse 2: Next generation datasets for self-driving perception and forecasting. *arXiv preprint arXiv:2301.00493*.

Wonsak, S., Al-Rifai, M., Nolting, M., and Nejdl, W. (2022). Multi-modal motion prediction with graphormers. In *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, pages 3521–3528. IEEE.

Wright, J. and Leyton-Brown, K. (2010). Beyond equilibrium: Predicting human behavior in normal-form games. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 24, pages 901–907.

Wu, H., Phong, T., Yu, C., Cai, P., Zheng, S., and Hsu, D. (2023a). What truly matters in trajectory prediction for autonomous driving? *arXiv preprint arXiv:2306.15136*.

Wu, P., Jia, X., Chen, L., Yan, J., Li, H., and Qiao, Y. (2022). Trajectory-guided control prediction for end-to-end autonomous driving: A simple yet strong baseline. *Advances in Neural Information Processing Systems*, **35**, 6119–6132.

Wu, P., Chen, L., Li, H., Jia, X., Yan, J., and Qiao, Y. (2023b). Policy pre-training for end-to-end autonomous driving via self-supervised geometric modeling. *arXiv preprint arXiv:2301.01006*.

Wulfmeier, M., Ondruska, P., and Posner, I. (2015). Maximum entropy deep inverse reinforcement learning. *arXiv preprint arXiv:1507.04888*.

Xi, W., Shi, L., and Cao, G. (2023a). An imitation learning method with data augmentation and post processing for planning in autonomous driving. https://opendrivelab.com/e2ead/AD23Challenge/Track_4_pe-gasus_weitao.pdf. Accessed: 2023-07-07.

Xi, W., Shi, L., and Cao, G. (2023b). An imitation learning method with data augmentation and post processing for planning in autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*.

Xu, D., Chen, Y., Ivanovic, B., and Pavone, M. (2022a). Bits: Bi-level imitation for traffic simulation. *arXiv preprint arXiv:2208.12403*.

Xu, H., Gao, Y., Yu, F., and Darrell, T. (2017). End-to-end learning of driving models from large-scale video datasets. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2174–2182.

Xu, J., Xiao, L., Zhao, D., Nie, Y., and Dai, B. (2022b). Trajectory prediction for autonomous driving with topometric map. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 8403–8408. IEEE.

Yao, W., Zhao, H., Bonnifait, P., and Zha, H. (2013). Lane change trajectory prediction by using recorded human driving data. In *2013 IEEE Intelligent vehicles symposium (IV)*, pages 430–436. IEEE.

Ye, M., Cao, T., and Chen, Q. (2021). Tpcn: Temporal point cloud networks for motion forecasting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11318–11327.

Ye, M., Xu, J., Xu, X., Wang, T., Cao, T., and Chen, Q. (2022). Dcms: Motion forecasting with dual consistency and multi-pseudo-target supervision. *arXiv preprint arXiv:2204.05859*.

Ye, T., Jing, W., Hu, C., Huang, S., Gao, L., Li, F., Wang, J., Guo, K., Xiao, W., Mao, W., Zheng, H., Li, K., Chen, J., and Yu, K. (2023). Fusionad: Multi-modality fusion for prediction and planning tasks of autonomous driving.

Yuan, Y. and Kitani, K. (2020). Dlow: Diversifying latent flows for diverse human motion prediction. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IX 16*, pages 346–364. Springer.

Yuan, Y., Weng, X., Ou, Y., and Kitani, K. M. (2021). Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9813–9823.

Yurtsever, E., Lambert, J., Carballo, A., and Takeda, K. (2020). A survey of autonomous driving: Common practices and emerging technologies. *IEEE access*, **8**, 58443–58469.

Zeng, W., Luo, W., Suo, S., Sadat, A., Yang, B., Casas, S., and Urtasun, R. (2019). End-to-end interpretable neural motion planner. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8660–8669.

Zeng, W., Wang, S., Liao, R., Chen, Y., Yang, B., and Urtasun, R. (2020). Dsdnet: Deep structured self-driving network. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXI 16*, pages 156–172. Springer.

Zeng, W., Liang, M., Liao, R., and Urtasun, R. (2021). Lanercnn: Distributed representations for graph-centric motion forecasting. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 532–539. IEEE.

Zhai, J.-T., Feng, Z., Du, J., Mao, Y., Liu, J.-J., Tan, Z., Zhang, Y., Ye, X., and Wang, J. (2023). Rethinking the open-loop evaluation of end-to-end autonomous driving in nuscenes. *arXiv preprint arXiv:2305.10430*.

Zhan, W., Liu, C., Chan, C.-Y., and Tomizuka, M. (2016). A non-conservatively defensive strategy for urban autonomous driving. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 459–464. IEEE.

Zhan, W., Sun, L., Wang, D., Shi, H., Clausse, A., Naumann, M., Kummerle, J., Konigshof, H., Stiller, C., de La Fortelle, A., *et al.* (2019). Interaction dataset: An international, adversarial and cooperative motion dataset in interactive driving scenarios with semantic maps. *arXiv preprint arXiv:1910.03088*.

Zhang, C., Guo, R., Zeng, W., Xiong, Y., Dai, B., Hu, R., Ren, M., and Urtasun, R. (2022a). Rethinking closed-loop training for autonomous driving. In *ECCV*, pages 264–282. Springer.

Zhang, K., Feng, X., Wu, L., and He, Z. (2022b). Trajectory prediction for autonomous driving using spatial-temporal graph attention transformer. *IEEE Transactions on Intelligent Transportation Systems*, **23**(11), 22343–22353.

Zhang, L., Ding, W., Chen, J., and Shen, S. (2020). Efficient uncertainty-aware decision-making for automated driving using guided branching. In *IEEE ICRA*, pages 3291–3297.

Zhang, L., Su, P.-H., Hoang, J., Haynes, G. C., and Marchetti-Bowick, M. (2021a). Map-adaptive goal-based trajectory prediction. In *Conference on Robot Learning*, pages 1371–1383. PMLR.

Zhang, Q., Hu, S., Sun, J., Chen, Q. A., and Mao, Z. M. (2022c). On adversarial robustness of trajectory prediction for autonomous vehicles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15159–15168.

Zhang, Y., Qian, D., Li, D., Pan, Y., Chen, Y., Liang, Z., Zhang, Z., Zhang, S., Li, H., Fu, M., *et al.* (2024). Graphad: Interaction scene graph for end-to-end autonomous driving. *arXiv:2403.19098*.

Zhang, Z., Liniger, A., Dai, D., Yu, F., and Van Gool, L. (2021b). End-to-end urban driving by imitating a reinforcement learning coach. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 15222–15232.

Zhang, Z., Liniger, A., Dai, D., Yu, F., and Van Gool, L. (2023). Trafficbots: Towards world models for autonomous driving simulation and motion prediction. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1522–1529. IEEE.

Zhao, A., He, T., Liang, Y., Huang, H., Van den Broeck, G., and Soatto, S. (2021a). Sam: Squeeze-and-mimic networks for conditional visual driving policy learning. In *Conference on Robot Learning*, pages 156–175. PMLR.

Zhao, H., Gao, J., Lan, T., Sun, C., Sapp, B., Varadarajan, B., Shen, Y., Shen, Y., Chai, Y., Schmid, C., *et al.* (2021b). Tnt: Target-driven trajectory prediction. In *Conference on Robot Learning*, pages 895–904. PMLR.

Zhao, T., Xu, Y., Monfort, M., Choi, W., Baker, C., Zhao, Y., Wang, Y., and Wu, Y. N. (2019). Multi-agent tensor fusion for contextual trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12126–12134.

Zheng, W., Song, R., Guo, X., and Chen, L. (2024). Genad: Generative end-to-end autonomous driving. *arXiv:2402.11502*.

Zhong, Z., Rempe, D., Xu, D., Chen, Y., Veer, S., Che, T., Ray, B., and Pavone, M. (2023). Guided conditional diffusion for controllable traffic simulation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3560–3566. IEEE.

Zhou, H., Li, W., Kong, Z., Guo, J., Zhang, Y., Yu, B., Zhang, L., and Liu, C. (2020). Deepbillboard: Systematic physical-world testing of autonomous driving systems. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, pages 347–358.

Zhou, Z., Ye, L., Wang, J., Wu, K., and Lu, K. (2022). Hivt: Hierarchical vector transformer for multi-agent motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8823–8833.

Ziebart, B. D., Maas, A. L., Bagnell, J. A., Dey, A. K., *et al.* (2008). Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pages 1433–1438. Chicago, IL, USA.

Ziegler, J., Bender, P., Dang, T., and Stiller, C. (2014). Trajectory planning for bertha—a local, continuous method. In *2014 IEEE intelligent vehicles symposium proceedings*, pages 450–457. IEEE.