2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

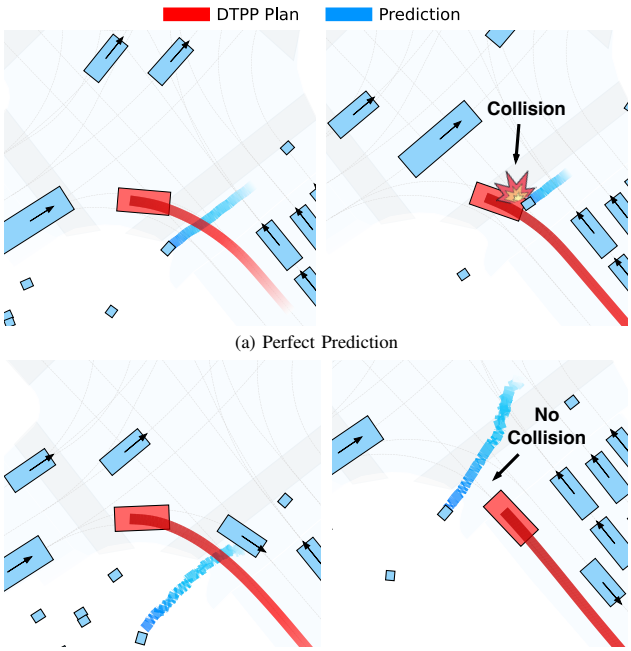
Perfect Prediction or Plenty of Proposals? What Matters Most in Planning for Autonomous Driving

Aron Distelzweig[1], Faris Janjoš[2], Oliver Scheel[2], Sirish Reddy Varra[2,3], Raghu Rajan[1], Joschka Boedecker[1]

***Abstract*— Traditionally, prediction and planning in autonomous driving (AD) have been treated as separate, sequential modules. Recently, there has been a growing shift towards tighter integration of these components, known as Integrated Prediction and Planning (IPP), with the aim of enabling more informed and adaptive decision-making. However, it remains unclear to what extent this integration actually improves planning performance. In this work, we investigate the role of prediction in IPP approaches, drawing on the widely adopted Val14 benchmark, which encompasses more common driving scenarios with relatively low interaction complexity, and the interPlan benchmark, which includes highly interactive and out-of-distribution driving situations. Our analysis reveals that even access to perfect future predictions does not lead to better planning outcomes, indicating that current IPP methods often fail to fully exploit future behavior information. Instead, we focus on high-quality proposal generation, while using predictions primarily for collision checks. We find that many imitation learning-based planners struggle to generate realistic and plausible proposals, performing worse than PDM—a simple lane-following approach. Motivated by this observation, we build on PDM with an enhanced proposal generation method, shifting the emphasis towards producing diverse but realistic and high-quality proposals. This proposal-centric approach significantly outperforms existing methods, especially in out-of-distribution and highly interactive settings, where it sets new state-of-the-art results.**

I. Introduction

Prediction plays a central role in the decision-making pipeline of autonomous driving systems. Accurately forecasting the behavior of surrounding agents is fundamental for understanding the dynamics of traffic scenes and for enabling safe and effective motion planning. In particular, predictions allow the ego vehicle to anticipate possible future developments in the environment and to adapt its maneuvers accordingly. Many existing systems [1]–[3] follow a modular architecture in which prediction and planning are treated as sequential stages: first, the future trajectories of other agents are predicted, then a separate planner generates a suitable trajectory for the ego vehicle based on these predictions. While this separation simplifies system design and modular evaluation, it limits the planner's ability to react to and influence the behavior of other agents, especially in dense or highly interactive traffic situations. In order to address this limitation, recent approaches [4]–[6] have begun to move towards a tighter integration of prediction and planning. Integrated Prediction and Planning (IPP) methods aim to reason jointly about the evolution of all traffic participants and the ego vehicle, leveraging mutual dependencies to improve decision-making. The underlying hypothesis is that a coupled system can more effectively handle complex interactions, resolve ambiguous situations, and produce more realistic and feasible plans. While one-stage approaches, whether end-to-end [1] or mid-to-mid [7], can be interpreted as forms of IPP, our focus here is on methods with more explicit and tighter integration of prediction and planning. However, the actual influence of prediction accuracy on planning performance in such integrated systems remains unclear. While more accurate predictions are assumed to translate into better planning outcomes in IPP methods, our results show that this relationship does not hold. Fig. 1 shows an example scenario where the state-of-the-art DTPP Planner [4] fails to benefit from perfect predictions, i.e., the true behavior of other agents under a certain plan. Its own predictions are inaccurate and unrealistic, however, they do not yield a collision. We have found similar behavior in other learned IPP approaches. This raises an intriguing question – **are**



(a) Perfect Prediction

(b) Learned Prediction from DTPP [4]

Fig. 1: Visualization of a real-world scenario from nuPlan Val14 dataset in reactive simulation. The ego vehicle attempts to cross an intersection while a pedestrian is crossing. Using the state-of-the-art DTPP [4] planner with **(a)** perfect predictions (i.e., the actual responses of agents under the given plan) results in a collision, whereas DTPP with **(b)** learned predictions successfully avoids the pedestrian. Even when provided with ground-truth future motion, a state-of-the-art planner generates a colliding trajectory. In contrast, it avoids collisions when relying on its own predictions, although these predictions are noisy and inaccurate.

[1]Department of Computer Science, University of Freiburg, Germany.
[2]Bosch Center for Artificial Intelligence, Germany.
[3]RWTH Aachen University, Germany.

planners properly using predictions?

In order to investigate this issue, we evaluate state-of-the-art IPP approaches under three different conditions: using perfect predictions, learned / rule-based predictions and no predictions. Our findings demonstrate that these methods generally fail to integrate predictions effectively, regardless of prediction accuracy. Moreover, widely used benchmarks such as the nuPlan datasets Test14 [8] and Val14 [9] do not sufficiently represent the complexity of highly interactive driving scenarios. These benchmarks are biased towards relatively static or straightforward situations, lacking sufficient representation of multi-agent coordination, negotiation, or conflict resolution. Consequently, IPP methods are frequently assessed in settings that do not rigorously challenge their interactive reasoning capabilities. We complement evaluations on the commonly used Val14 benchmark with evaluations on the more challenging interPlan [10] benchmark, which emphasizes highly interactive scenarios.

Additionally, our analysis reveals a critical limitation of many imitation learning-based planning approaches: their tendency to generate trajectories that are neither dynamically feasible nor behaviorally realistic, see Fig. 2 for an example. This limitation is evident not only in complex, rare-event interactions requiring advanced anticipatory behaviors as shown in Fig. 2, but also in simpler, more common driving contexts. Our findings lead us to a shift in emphasis: focus on **generating strong proposals**, and rely on predictions for collision avoidance during test-time. To this end, we introduce a rule-based proposal generation method that explicitly models a wide range of realistic maneuvers, including rare and long-tail behaviors. We demonstrate that incorporating such proposals leads to significant performance gains across a range of interactive and out-of-distribution scenarios, outperforming all existing approaches by a large margin. In summary, we:

1) Show that perfect predictions in existing IPP approaches do not lead to better planning performance.
2) Demonstrate that current state-of-the-art planners are not able to generate high-quality ego proposals.
3) Introduce a novel flexible rule-based planner that is able to generate more complex and diverse driving maneuvers than existing planners.
4) Set new state-of-the-art results on long-tail scenarios in interPlan and outperform all existing approaches by a large margin.

II. Related Work

The following sections provide an overview of related work in prediction, planning, and their integration.

A. Prediction

Prediction plays a fundamental role in autonomous driving, enabling the vehicle to anticipate the dynamics of its surrounding environment. Broadly, prediction models address two core tasks: (1) context encoding, which involves capturing scene structure and dynamics, and (2) future uncertainty modeling, which accounts for the multi-modal nature of agent behavior. Early prediction models [14]–[16] often utilized grid-based scene representations, which, despite their effectiveness, suffered from limited spatial resolution. More recent methods [17], [18] adopt vectorized representations, encoding elements such as lanes, agent trajectories, and road boundaries in a structured format, allowing for richer and more scalable modeling of complex environments.

To account for the multi-modal nature of agent behavior, prediction models typically generate multiple future trajectories. This can be achieved by generating a set of discrete trajectories [18], conditioning predictions on different map elements (e.g., lanes or goals) [19], or using latent variable models such as CVAEs [20]–[23] or diffusion models [24], where different latent samples produce diverse plausible futures. Recent works like [13], [25] frame trajectory prediction as a next-token language modeling task using discrete representations.

B. Planning

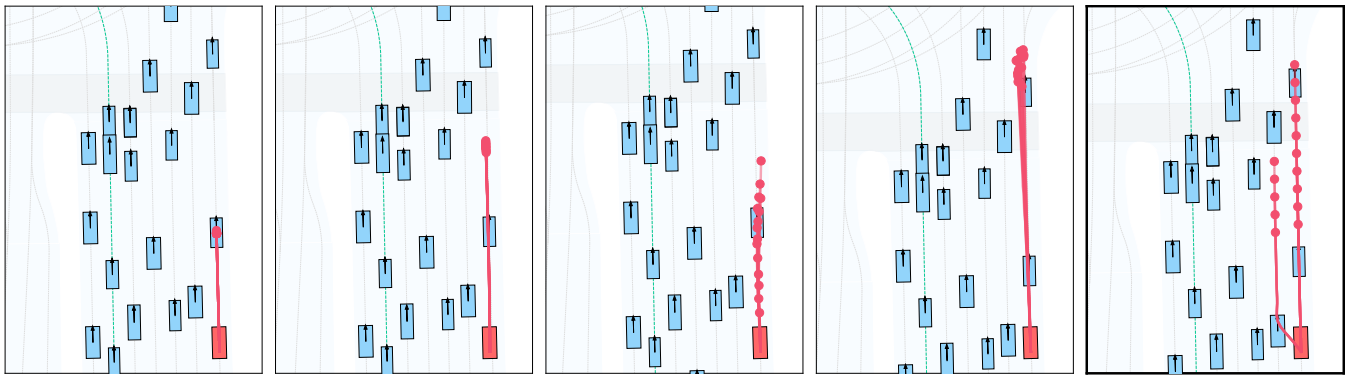
Planning in autonomous driving involves generating feasible and safe trajectories that navigate towards a goal while accounting for dynamic interactions with the environment. Planning approaches can be broadly categorized into rule-based and learning-based methods.

Rule-based planners rely on heuristics or physics models, offering interpretability and often safety guarantees. A widely used method is the Intelligent Driver Model (IDM) [26], which models car-following behavior by maintaining a safe distance from the leading vehicle. Extensions like the Predictive Driver Model (PDM) [9] incorporate elements of Model Predictive Control (MPC), generating candidate trajectories and selecting the best one based on a cost function.

Learning-based planners learn driving policies from data and are typically categorized into Imitation Learning (IL) and Reinforcement Learning (RL). IL uses supervised learning to mimic expert behavior. However, IL suffers from covariate shift, leading to compounding errors [27], [28]. ChauffeurNet [7] mitigates this issue by injecting perturbations to expert demonstrations, exposing the model to off-distribution states. Recent advances explore transformer-based [29] and diffusion-based [11], [30] architectures for motion planning. Goal-Conditioned PGP (GC-PGP) [12] extends the internal graph-based scene representation of the prediction model PGP [19], enabling goal-conditioning for planning. RL methods formulate planning as a Markov Decision Process (MDP) and optimize driving policies through interaction. These methods have been applied to lane changing [31], car following [32], overtaking [33], and real-world driving [34] tasks. Despite its potential, RL presents challenges including sample inefficiency, safety risks during exploration, and the difficulty of reward function design, although recent advances such as GPUDrive [35] offer a promising direction to address these limitations.

C. Integrated Prediction and Planning

Although traditionally treated as separate modules, decoupling prediction and planning can lead to suboptimal behavior, making integration essential for socially-aware decision-



(a) GameFormer [5] (b) DiffusionPlanner [11] (c) GC-PGP [12] (d) AR GameFormer [5], [13] (e) SPDM (Ours)

Fig. 2: Visualization of $N_p = 15$ ego trajectory proposals generated by various approaches. Each proposal's endpoint is indicated by a red dot. Subfigures (a)–(d) present proposals from learning-based methods, while subfigure (e) illustrates proposals from our rule-based method. In this highly interactive scenario, the ego vehicle must perform multiple consecutive lane changes to reach the designated goal lane, highlighted in green. Among the approaches presented, only our proposed method considers a lane change in this scenario.

making. A central challenge in this process is effectively incorporating predictions into the planning pipeline. According to [36], integration strategies can be categorized into five variants: *human leader*, *robot leader*, *implicit*, *joint*, and *co-leader*. Most existing systems adopt a sequential human leader scheme [1], [2], [6], [37], in which the ego vehicle's trajectory is conditioned on the predicted motions of other agents. However, this one-way dependency can result in overly conservative behavior [38].

To address these limitations, recent research has explored tighter integration through joint reasoning or bidirectional coupling. In joint approaches [5], [39], [40], the ego and surrounding agents are modeled simultaneously, with their trajectories optimized under a shared global objective. For example, GameFormer [5] frames the planning problem as a hierarchical game using a transformer-based architecture that iteratively refines the interactions. However, these methods may be limited by their assumption of shared intent, which may not hold in real-world traffic. In contrast, bidirectional integration methods such as co-leader approaches model the mutual influence between the ego's plan and predicted behaviors of others, without assuming a shared objective, which makes them more flexible in capturing interactive behaviors. [38] proposes a method that employs a learned behavioral model to stochastically forecast multi-agent trajectories, combined with a planning objective to compute contingency plans. In contrast, [4], [41] formulate planning as a discrete MDP with two trees, representing ego and other agents' potential behaviors, solved via dynamic programming.

End-to-End approaches [1], [42] fall into the broader category of IPP, however, our focus is on advanced integration methods, which can often be embedded within End-to-End frameworks, so the methods we study can be seen as specific building blocks of the broader End-to-End paradigm.

D. Benchmarks

In recent years, several large-scale datasets and simulation frameworks have been developed to standardize the evaluation of autonomous driving systems. Some benchmarks focus primarily on open-loop prediction [43]–[45], while others support closed-loop planning evaluation [46], [47]. Given the interdependence of prediction and planning, it is increasingly recognized that IPP systems should be evaluated as a whole. nuPlan [47] is one of the most established and relevant frameworks for planning. It provides a simulation framework built on real-world driving logs and supports open-loop and closed-loop evaluation settings. In the non-reactive setting, the surrounding agents follow recorded trajectories from the dataset. In the reactive setting, their behavior is modeled using Intelligent Driver Model (IDM) [26], allowing for limited interaction with the ego. The Val14 benchmark is the standardized evaluation split for nuPlan, but it primarily consists of straightforward, low-interaction scenarios that can often be solved with basic lane-following behavior. To address this limitation, interPlan [10] augments the nuPlan dataset by introducing rare long-tail scenarios through manual scene modification.

III. METHODOLOGY

To systematically investigate Integrated Prediction and Planning (IPP) approaches, we select a set of representative and widely recognized methods from the literature. These include: Differentiable Tree Policy Planning (DTPP) [4], a tree-based co-leader strategy; GameFormer [5], which jointly models multi-agent interactions through a game-theoretic framework; and Differentiable Integrated Prediction and Planning (DIPP) [6], a non-linear optimization-based method that jointly incorporates prediction into the planning process.

Section III-A provides an overview of how each selected approach integrates prediction and planning. For in-depth methodological details, we refer the reader to the respective original work. Section III-B presents the methodology employed to generate perfect prediction models across different simulation settings. Finally, Section III-C presents our proposed proposal generation method.

A. Integrated Prediction and Planning Approaches

The following section provides a brief overview of the three IPP approaches: DTPP, GameFormer, and DIPP with a particular emphasis on how each method integrates prediction and planning.

Differentiable Tree Policy Planning (DTPP) [4] constructs a trajectory tree through iterative node expansion, guided by learned ego-conditioned prediction and cost models, until a predefined maximum number of expansion stages, N_l, is reached. At each stage k, every branch of the ego trajectory tree is expanded by a rule-based proposal mechanism. The ego-conditioned prediction model then forecasts the future trajectories of surrounding agents conditioned on these ego proposals. The prediction model follows a Transformer encoder–decoder architecture. The encoder maps agent histories $A \in \mathbb{R}^{N_a \times T_h \times d_a}$, where N_a is the number of surrounding agents, T_h the history length, and d_a the agent attributes, together with map features $M \in \mathbb{R}^{N_m \times N_p \times d_m}$, where N_m is the number of map elements, N_p the number of waypoints, and d_m the map attributes, into a scene context $C = \mathbf{Enc}(A, M)$. The decoder then produces ego-conditioned predictions $Y^{(k)} = \mathbf{Dec}(C, E^{(k)}) \in \mathbb{R}^{B \times N_a \times T_f^k \times 2}$ conditioned on the scene context and the ego trajectory tree $E^{(k)} \in \mathbb{R}^{B \times T_f^k \times 3}$, where B is the number of branches and T_f^k the future length at stage k. Based on these interaction-aware predictions, a learned cost function evaluates and ranks ego trajectory candidates $s = c(E^{(k)}, Y^{(k)})$, selecting the most promising ones for further expansion. The cost function is optimized via Inverse Reinforcement Learning (IRL). In practice, the number of expansion stages is set to $N_l = 2$.

GameFormer [5] unifies prediction and planning within a game-theoretic framework. Agent histories A and map features M are first encoded into a scene context representation $C = \mathbf{Enc}(A, M)$ using a Transformer encoder. On top of this context, N_l Transformer-based decoders model multi-agent interactions across different reasoning levels. At level-0, the decoder jointly predicts the future trajectories of all agents, including the ego vehicle, $Y_e^{(0)} = \mathbf{Dec}^{(0)}(C) \in \mathbb{R}^{N_a \times T_f \times 2}$, where modality embeddings and agent histories serve as queries and the scene context C provides the keys and values. We denote $Y_e^{(k)} \in \mathbb{R}^{(N_a+1) \times T_f \times 2}$ as the predictions at level-k for all agents including the ego agent. At level-k ($k \geq 1$), the decoder refines predictions by incorporating the trajectories from the previous level, $Y_e^{(k)} = \mathbf{Dec}^{(k)}(C, Y_e^{(k-1)})$. Through self-attention, each agent attends to the trajectories of all other agents while masking its own, thereby modeling interactive responses. Predictions are iteratively refined over $N_l = 3$ levels, resulting in a hierarchy of interaction-aware future trajectories.

Differentiable Integrated Prediction and Planning (DIPP) [6] employs a neural network architecture to jointly predict the future trajectories of surrounding agents, denoted as $Y_{e,i\geq 1}$, and an initial plan for the ego vehicle, denoted as $Y_{e,i=0}$. At its core lies a differentiable optimization module that refines the initial ego plan by accounting for the predicted behaviors of other agents and optimizing multiple objectives, including compliance with speed limits, passenger comfort, adherence to traffic rules, and safety, measured in terms of collision avoidance and maintaining appropriate distances to other vehicles. The optimization problem is formulated as $u^* = \arg\min_u \frac{1}{2} \sum_i \|\omega_i \, c_i(u, Y_{e,i\geq 1})\|^2$, where $u = \{u_1, u_2, \ldots, u_T\}$ are the control inputs to be optimized, c_i are cost terms, and ω_i are their corresponding weights. The predicted initial plan $Y_{e,i=0}$ serves as initialization for the control inputs. In contrast to the original work, we employ GameFormer with $N_l = 0$ levels to predict the surrounding agents' future trajectories as well as the ego's initial plan. Furthermore, we integrate GameFormer with $N_l = 3$ levels into the DIPP framework, denoted as *GameFormer + DIPP*.

B. Prediction Modalities in IPP Methods

To examine the impact of predictions on planning performance, we systematically vary the type of predictions used in the selected IPP methods. Let $Y \in \mathbb{R}^{N_a \times T_f \times 2}$ be the x, y positions of N_a neighboring agents for T_f future timesteps. In our experiments, we set the number of agents, $N_a = 10$, corresponding to the nearest agents to the ego vehicle. We consider the following conditions:

1) Learned / Rule-based Predictions: Set $Y = \hat{Y}$, where $\hat{Y} \in \mathbb{R}^{N_a \times T_f \times 2}$ are predictions generated by a learned or rule-based prediction module.

2) Perfect Predictions: The actual future behavior of other agents is provided, i.e., $Y = Y^*$, where $Y^* \in \mathbb{R}^{N_a \times T_f \times 2}$ denotes the ground-truth trajectories of surrounding agents. In the non-reactive closed-loop and open-loop setting, Y^* corresponds to recorded future trajectories available in the dataset. For reactive simulations, we utilize the Intelligent Driver Model (IDM) [26] to generate behavior that responds to the actions of the ego vehicle. Since pedestrians are not simulated via IDM in nuPlan, their recorded future behavior is used instead, even in reactive settings.

3) No Predictions: no predictions are made about the future behavior of other agents; we mask out the futures of all agents by $Y = \mathbf{0}$.

We implement the conditions above in a manner tailored to each model. In DTPP, predictions are replaced at every stage whereas in GameFormer they are replaced in each decoder. In DIPP, predictions are replaced before the refinement procedure.

C. Spline-Fitting PDM (SPDM)

Our method for generating improved ego proposals builds upon the PDM-Closed approach and the proposal generation strategy from [4]. PDM-Closed generates IDM-based proposals along the centerline with varying target velocities, while assuming constant velocity for the other agents. It then selects the optimal proposal using a scoring function closely aligned with the nuPlan metric score. Our approach comprises two stages. In the first stage, following the PDM methodology, we generate IDM proposals along the current centerline by varying the target velocity across five values, $\{20, 40, 60, 80, 100\}\%$ of the current speed limit, and applying three lateral offsets, $\{-1, 0, +1\}$m.

Model	Predictions	Val14			interPlan
		NR↑	R↑	OL↑	R↑
PDM-Closed [9]	Constant Velocity	92.84	92.28	42.52	41.88
	Perfect	**93.80**	N/A	**46.96**	N/A
	None	41.07	40.33	15.74	30.63
DTPP [4]	Learned	**64.54**	**68.23**	**67.05**	**37.27**
	Perfect	53.65	64.04	32.52	25.12
	None	51.09	54.81	56.49	21.72
GameFormer [5]	Learned	**34.99**	**34.98**	**76.33**	**4.10**
	Perfect	33.92	34.82	76.31	3.28
	None	30.24	26.71	76.26	3.28
GameFormer [5] + DIPP [6]	Learned	79.83	80.95	80.38	**21.26**
	Perfect	**81.04**	**81.15**	**81.01**	19.00
	None	58.14	51.14	80.73	13.19
DIPP [6]	Learned	73.35	77.65	37.21	**21.25**
	Perfect	**74.19**	**79.02**	**37.67**	12.90
	None	49.58	56.86	37.25	13.07

TABLE I: Performance of various integrated prediction and planning approaches on the Val14 and interPlan benchmarks, evaluated under different prediction sources: learned/rule-based, perfect predictions, or no predictions. NR/R: non-reactive/reactive simulation and OL: open-loop simulation.

Our spline-fitting extension expands the set by including trajectories that are not constrained to the current centerline. To this end, we extract all lanes along the goal-directed route, referred to as route lanes, and generate additional proposals for each route lane using the same target speed variations as in the first stage. We model the velocity profile over time between the current time and the time horizon, T_f. Let the initial ego state be defined as $s(0) = [x_0, y_0, v_0, a_0, \theta_0, l_0]$, where the components denote position coordinates, velocity, acceleration, heading angle, and lane position, respectively. For each target velocity, we fit a cubic polynomial to describe the velocity profile over time t:

$$v(t) = c_0 + c_1 t + c_2 t^2 + c_3 t^3, \tag{1}$$

$$v(0) = v_0, \quad v(T_f) = v_{\text{target}}, \tag{2}$$

$$\dot{v}(0) = a_0, \quad \dot{v}(T_f) = 0. \tag{3}$$

Here, (c_0, c_1, c_2, c_3) are polynomial coefficients. This formulation ensures smooth transitions in both velocity and acceleration, thereby enabling the generation of realistic and diverse ego trajectories that are not restricted to a single lane. The proposals generated from both stages are merged into a unified candidate set. Following the PDM procedure, control actions are then iteratively computed using a Linear Quadratic Regulator (LQR) controller, and the ego state is propagated using a kinematic bicycle model [48]. All candidate proposals are then evaluated using the PDM scoring function inspired by the nuPlan metrics. This function considers factors such as time-to-collision, collisions, progress, comfort, drivable area adherence, and compliance with driving direction. The proposal with the highest score is chosen as the final plan. Overall, our SPDM method is lightweight, produces realistic and kinematically feasible trajectories, and can be seamlessly integrated into existing motion generation pipelines.

IV. Experiments

A. Evaluation

We employ the nuPlan framework [47], a closed-loop simulator based on real-world driving data. For evaluation, we use the Val14 benchmark [9], which comprises 1,118 scenarios from the nuPlan validation set, covering up to 100 examples per type across 14 scenario categories. We further consider the Test14 benchmark [8], which is divided into Test14-random and Test14-hard. Test14-random consists of randomly sampled scenarios from each category, whereas Test14-hard is constructed by selecting the 20 most challenging cases per type based on PDM-Closed [9] performance. Additionally, we consider the recently introduced out-of-distribution interPlan benchmark [10], which consists of 80 complex driving scenarios. Both benchmarks simulate background traffic using IDM in reactive closed-loop setting.

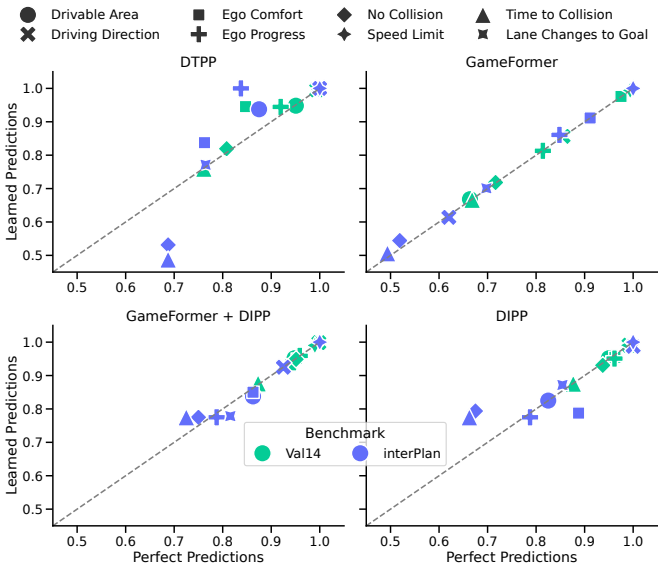


Fig. 3: Comparative analysis of subscores between learned and perfect predictions when utilized within DTPP, GameFormer, GameFormer + DIPP, and DIPP across the Val14 and interPlan benchmark in reactive simulation. Markers in the upper triangle denote better performance with learned predictions, whereas markers in the lower triangle denote better performance with perfect predictions. Subscore 'Lane Changes to Goal' is exclusively available in the interPlan benchmark.

Type	Proposal Generator	N_p	Val14		interPlan
			NR↑	R↑	R↑
Rule-based	PDM-Closed [9]	15	**92.84**	**92.28**	41.88
Diffusion	Diffusion-ES* [30]	15	75.98	79.92	**57.41**
	Diffusion-ES [30]	15	72.77	75.80	35.18
	DiffusionPlanner [11]	15	91.26	91.13	25.76
NAR Transformer	GameFormer [5] + DIPP [6]	15	82.98	83.39	19.09
	GameFormer [5]	15	42.95	40.30	3.95
AR Transformer	AR GameFormer [5], [13]	15	43.46	44.22	1.02
Graph-based	GC-PGP [12]	15	58.40	63.82	14.55

TABLE II: PDM planner performance on the Val14 and interPlan benchmarks across a diverse set of proposal generators, with the best proposal selected using the PDM scoring function. Our comparison spans a broad range of model families, including rule-based, diffusion, transformer-based, and graph-based approaches. Diffusion-ES* uses PDM proposals as initial set instead of learned proposals. NR/R: non-reactive/reactive simulation

B. Perfect Predictions Do Not Improve Planning

To assess whether IPP methods utilize predictions in a meaningful manner, we systematically modified the predictions within these approaches, as outlined in Section III-

B. The outcomes of this analysis are presented in Table I. PDM-Closed follows a sequential integration of predictions and, therefore, cannot be evaluated with perfect predictions in reactive closed-loop simulations. This constitutes a clear limitation, as other agents dynamically adapt their behavior in response to the ego agent, rendering fixed predictions inherently suboptimal. Nonetheless, we include PDM-Closed as a baseline. The results reveal a consistent trend: all evaluated IPP methods fail to effectively leverage predictions. Even when provided with perfect predictions, none of the approaches demonstrate a significant improvement in planning performance. Notably, the co-leader method DTPP exhibits a decline in performance across all simulation settings compared to when including learned predictions. GameFormer and DIPP exhibit comparable performance when utilizing perfect predictions compared to their performance with learned predictions on the Val14 benchmark. However, on the interPlan benchmark, both methods show a reduction in performance when provided with perfect predictions.

Across all methods, a complete removal of predictions results in a decrease in performance, indicating that knowledge of future scene dynamics generally contributes to improved planning outcomes. However, it is counterintuitive that the use of perfect predictions does not yield a significant performance gain. One could argue that this results from the distributional shift between real-world driving and simulation. However, we also observe no improvement when using ground-truth data in the non-reactive setting, indicating that even with in-distribution data, perfect predictions do not yield performance gains. In order to assess whether access to perfect predictions yields improvements in specific subscores, we decompose the final evaluation scores into their constituent subscores, as shown in Figure 3. The analysis indicates that only DTPP exhibits a significant improvement in both collision rate and time-to-collision on the interPlan benchmark, while none of the other approaches show any statistically significant improvement. Interestingly, DTPP exhibits only a marginal performance drop on the Val14 benchmark. This observation is consistent with the fact that Val14 scenarios lack high interactivity. GameFormer and GameFormer + DIPP demonstrate comparable performance under both learned and perfect prediction settings. Notably, DIPP shows a degradation in collision-related metrics when using perfect predictions on the interPlan benchmark. These findings suggest that, among the evaluated methods, only DTPP exhibits a limited ability to leverage perfect predictions for reducing collisions. However, the overall limited improvement underscores a substantial gap in the effective utilization of predictions. Despite having access to the true behavior of other agents, these methods continue to exhibit critical failure cases, thereby revealing a significant limitation in their capacity to exploit predictions.

C. Learning-based Approaches Are Ineffective Proposal Generators

A critical component of IPP methods is the plan generation process, which can be broadly categorized into rule-based and learning-based approaches. Rule-based methods typically generate multiple candidate plans from which the most suitable one is selected. In contrast, learning-based approaches aim to directly produce an optimal plan in a single inference step, offering greater adaptability and efficiency.

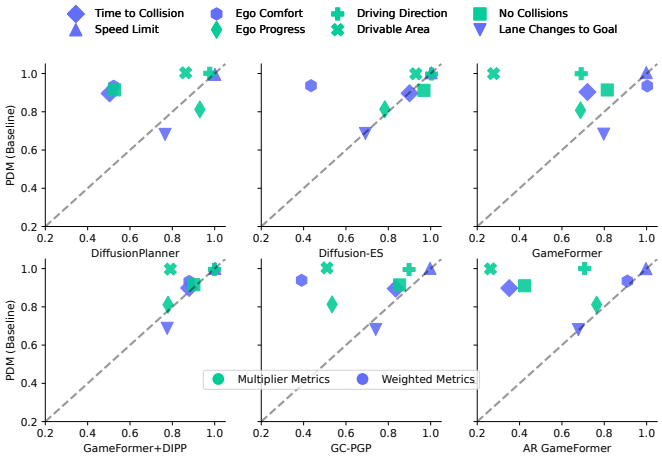


Fig. 4: Subscore performance comparison between learned-based proposal generators and the rule-based PDM-Closed on the interPlan benchmark in reactive simulation. Each point represents a specific subscore, with the y-axis indicating PDM-Closed performance and the x-axis showing performance of the corresponding learned-based generator. For the learned-based methods, the best proposal is selected using the PDM scoring function.

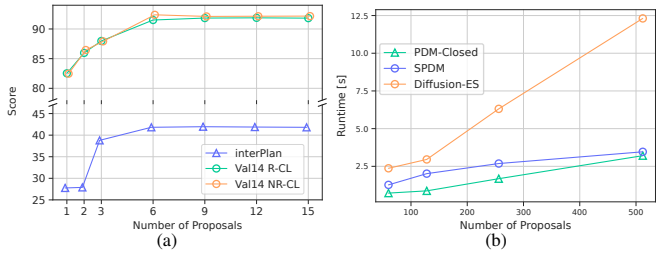


Fig. 5: Figure (a) shows performance of PDM across varying numbers of proposals on the Val14 and interPlan benchmarks. NR/R: non-reactive/reactive simulation. Figure (b) shows runtime analysis across varying numbers of proposals, averaged over 10 distinct scenarios.

We aim to investigate whether learning-based approaches can be used as proposal generators. To this end, we compare learning-based planners with the rule-based PDM planner. However, our observations indicate that existing imitation learning-based planners often exhibit significant limitations in planning efficacy. Specifically, they consistently fail to produce trajectories that yield high-quality driving performance. To empirically support this claim, we selected a set of state-of-the-art learning-based approaches from diverse model families and used each to generate multiple trajectory proposals. We then evaluate these proposals using a scoring function closely aligned with the nuPlan metric and select the highest-scoring proposal as the final plan. Table II presents the results for $N_p = 15$ proposals generated per method. For GameFormer [5] and GC-PGP [12], we set the number of modes to $m = 15$ and retrain the models accordingly. In the case of DiffusionPlanner [11], we set the sampling temperature to $t = 1.5$ to encourage greater diversity in

Proposal Generator	N_p	Val14		interPlan
		NR↑	R↑	R↑
PDM-Closed [9]	60	91.21	91.26	42.96
SPDM (Ours)	15	**92.84**	92.28	42.00
	20	92.73	92.21	49.72
	25	92.71	92.26	52.77
	30	92.71	92.44	59.07
	45	92.75	92.51	62.37
	60	92.75	**92.72**	**63.66**

TABLE III: Performance analysis of the proposed spline-fitting PDM method across different numbers of proposals N_p on the Val14 and interPlan benchmarks. NR/R: non-reactive/reactive simulation.

Method	R ↑			
	Val14	Test14-hard	Test14-random	interPlan
Urban Driver [29]	64.11	49.95	67.15	4.00
GC-PGP [12]	63.82	39.36	51.39	14.55
PlanCNN [50]	72.00	52.20	67.50	—
PlanTF [8]	76.95	61.61	79.58	30.53
PDM-Open [9]	54.24	35.83	57.23	—
PDM-Closed [9]	92.28	75.19	91.63	41.88
GameFormer [5]	79.78	67.05	82.05	21.26
DiffusionPlanner [11]	82.80	69.22	82.93	25.76
DTPP [4]	66.09	63.66	—	30.32
SPDM (Ours)	**92.72**	**80.75**	**92.54**	**63.66**

TABLE IV: Performance comparison on the nuPlan benchmarks in the reactive (R) simulation. Results are reported for Val14, Test14-hard, Test14-random, and interPlan, with higher scores indicating better closed-loop performance.

the generated trajectories. Diffusion-ES [30] combines PDM-generated proposals with additional learned ones. However, to ensure fair comparison across methods, we also evaluate a variant that excludes the rule-based PDM proposals from the initial candidate set. The autoregressive transformer-based encoder–decoder architecture constitutes an autoregressive adaptation of the GameFormer architecture. For tokenization, we follow the approach proposed in [13], and generate diverse proposals using nucleus sampling [49]. The findings indicate that proposals from PDM significantly outperform most learning-based approaches, particularly on the interPlan benchmark, where the performance gap is most pronounced. An exception is Diffusion-ES*, which achieves strong results, likely due to its initialization with PDM proposals. Notably, all evaluated learning-based planners were trained using motion forecasting objectives via imitation learning.

In Figure 4, we present a detailed comparison of subscore metrics between PDM and the highest-quality proposal among the $N_p = 15$ generated by each respective learning-based method. Across most subscore categories, PDM significantly outperforms the learning-based baselines. Prior work [9] has already highlighted a fundamental misalignment between the objectives of motion forecasting and planning. Our comprehensive evaluation reveals that purely imitation learning-based approaches consistently fail to produce good plans, not only in rare or out-of-distribution scenarios as in the interPlan benchmark, but also in common and relatively simple situations as in the Val14 benchmark.

D. High Quality Proposals Improve Planning

Building on the findings presented in Section IV-B, we aim to further investigate the underlying causes of the observed results. While some performance limitations may arise from the characteristics of individual planning approaches, such as the learned cost function in DTPP, in our analysis, we intentionally avoid attributing these limitations to specific methods. Instead, we hypothesize the existence of a shared structural bottleneck across planning approaches. Specifically, we question whether improvements in predictions can meaningfully enhance planning performance, or whether advancements must instead stem from the plan generation process itself. As demonstrated in Section IV-C, learning-based methods have shown limited effectiveness in generating high-quality plans. Consequently, we shift our focus to rule-based planning strategies. In an initial experiment, we evaluate whether the PDM proposals can contribute to improved driving performance.

Figure 5a illustrates that decreasing the number of proposals (originally set to $N_p = 15$) results in a performance drop when fewer than $N_p = 6$ proposals are used. Notably, even with as few as $N_p = 3$, the performance still surpasses that of most learning-based planners on both the Val14 and interPlan benchmarks. Furthermore, increasing N_p beyond the threshold value of 6 does not yield additional performance gains, indicating a saturation point for the benefit of simple trajectory diversification. To address the limitations of PDM in complex scenarios, we introduced the SPDM approach in Section III-C, designed to generate more sophisticated driving maneuvers. As shown in Table III, the inclusion of SPDM generated proposals results in a 50% improvement in driving performance on the interPlan benchmark compared to the original PDM proposal set. In contrast, increasing the number of PDM proposals alone does not yield additional performance gains. A similar trend is observed for SPDM on the Val14 benchmark. Finally, we show that our approach also is compute-efficient. Figure 5b presents a runtime comparison between PDM-Closed, Diffusion-ES, and our proposed method using different numbers of proposals. We include only Diffusion-ES in this comparison, as it represents the best-performing learning-based baseline on interPlan. Our method shows a slightly higher computational cost compared to PDM-Closed when using a small number of proposals, mainly because of the additional processing needed to incorporate neighboring lanes. However, as the number of proposals increases, the runtime becomes comparable to that of PDM-Closed. This is because PDM-Closed adapts proposals to predictions, while our method generates proposals independently, making the proposal generation process more efficient. In contrast, Diffusion-ES demonstrates significantly lower efficiency, with a large increase in runtime compared to SPDM and PDM-Closed. In Table IV, we present a performance comparison across multiple benchmarks, demonstrating that our method achieves superior performance, especially in more complex scenarios.

V. Conclusion

In this work, we demonstrated that current Integrated Prediction and Planning (IPP) approaches face substantial challenges in effectively leveraging predictions, with many existing planning methods further constrained by their limited ability to generate high-quality trajectories. In contrast, our proposed proposal generation method delivers a notable 50% performance improvement in out-of-distribution scenarios. These findings highlight a critical need to rethink how prediction is integrated into planning and to advance proposal generation as a central component of future IPP systems.

References

[1] Y. Hu, J. Yang, L. Chen, K. Li, C. Sima, X. Zhu, S. Chai, S. Du, T. Lin, W. Wang, L. Lu, X. Jia, Q. Liu, J. Dai, Y. Qiao, and H. Li, "Planning-oriented autonomous driving," 2023.

[2] W. Zeng, S. Wang, R. Liao, Y. Chen, B. Yang, and R. Urtasun, "Dsdnet: Deep structured self-driving network," 2020.

[3] A. Sadat, S. Casas, M. Ren, X. Wu, P. Dhawan, and R. Urtasun, "Perceive, predict, and plan: Safe motion planning through interpretable semantic representations," 2020.

[4] Z. Huang, P. Karkus, B. Ivanovic, Y. Chen, M. Pavone, and C. Lv, "Dtpp: Differentiable joint conditional prediction and cost evaluation for tree policy planning in autonomous driving," 2024.

[5] Z. Huang, H. Liu, and C. Lv, "Gameformer: Game-theoretic modeling and learning of transformer-based interactive prediction and planning for autonomous driving," 2023.

[6] Z. Huang, H. Liu, J. Wu, and C. Lv, "Differentiable integrated motion prediction and planning with learnable cost function for autonomous driving," 2023.

[7] M. Bansal, A. Krizhevsky, and A. Ogale, "Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst," 2018.

[8] J. Cheng, Y. Chen, X. Mei, B. Yang, B. Li, and M. Liu, "Rethinking imitation-based planners for autonomous driving," 2023.

[9] D. Dauner, M. Hallgarten, A. Geiger, and K. Chitta, "Parting with misconceptions about learning-based vehicle motion planning," 2023.

[10] M. Hallgarten, J. Zapata, M. Stoll, K. Renz, and A. Zell, "Can vehicle motion planning generalize to realistic long-tail scenarios?" 2024.

[11] Y. Zheng, R. Liang, K. Zheng, J. Zheng, L. Mao, J. Li, W. Gu, R. Ai, S. E. Li, X. Zhan, and J. Liu, "Diffusion-based planning for autonomous driving with flexible guidance," 2025.

[12] M. Hallgarten, M. Stoll, and A. Zell, "From prediction to planning with goal conditioned lane graph traversals," 2023.

[13] J. Philion, X. B. Peng, and S. Fidler, "Trajeglish: Traffic modeling as next-token prediction," 2024.

[14] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. S. Torr, and M. Chandraker, "Desire: Distant future prediction in dynamic scenes with interacting agents," 2017.

[15] H. Cui, V. Radosavljevic, F.-C. Chou, T.-H. Lin, T. Nguyen, T.-K. Huang, J. Schneider, and N. Djuric, "Multimodal trajectory predictions for autonomous driving using deep convolutional networks," 2019.

[16] S. Casas, W. Luo, and R. Urtasun, "Intentnet: Learning to predict intention from raw sensor data," 2021.

[17] J. Gao, C. Sun, H. Zhao, Y. Shen, D. Anguelov, C. Li, and C. Schmid, "Vectornet: Encoding hd maps and agent dynamics from vectorized representation," 2020.

[18] M. Liang, B. Yang, R. Hu, Y. Chen, R. Liao, S. Feng, and R. Urtasun, "Learning lane graph representations for motion forecasting," 2020.

[19] N. Deo, E. Wolff, and O. Beijbom, "Multimodal trajectory prediction conditioned on lane-graph traversals," 2022.

[20] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data," 2021.

[21] A. Cui, S. Casas, A. Sadat, R. Liao, and R. Urtasun, "Lookout: Diverse multi-future prediction and planning for self-driving," 2021.

[22] Y. Yuan, X. Weng, Y. Ou, and K. Kitani, "Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting," 2021.

[23] F. Janjoš, M. Hallgarten, A. Knittel, M. Dolgov, A. Zell, and J. M. Zöllner, "Conditional unscented autoencoders for trajectory prediction," 2024.

[24] C. M. Jiang, A. Cornman, C. Park, B. Sapp, Y. Zhou, and D. Anguelov, "Motiondiffuser: Controllable multi-agent motion prediction using diffusion," 2023.

[25] A. Seff, B. Cera, D. Chen, M. Ng, A. Zhou, N. Nayakanti, K. S. Refaat, R. Al-Rfou, and B. Sapp, "Motionlm: Multi-agent motion forecasting as language modeling," 2023.

[26] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," 2000.

[27] S. Ross, G. J. Gordon, and J. A. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," 2011.

[28] S. Hagedorn, A. Distelzweig, M. Hallgarten, and A. P. Condurache, "Learning through retrospection: Improving trajectory prediction for automated driving with error feedback," 2025.

[29] O. Scheel, L. Bergamini, M. Wołczyk, B. Osiński, and P. Ondruska, "Urban driver: Learning to drive from real-world demonstrations using policy gradients," 2021.

[30] B. Yang, H. Su, N. Gkanatsios, T.-W. Ke, A. Jain, J. Schneider, and K. Fragkiadaki, "Diffusion-es: Gradient-free planning with diffusion for autonomous driving and zero-shot instruction following," 2024.

[31] A. Alizadeh, M. Moghadam, Y. Bicer, N. K. Ure, U. Yavas, and C. Kurtulus, "Automated lane change decision making using deep reinforcement learning in dynamic and uncertain highway environment," 2019.

[32] M. Zhu, X. Wang, and Y. Wang, "Human-like autonomous car-following model with deep reinforcement learning," 2019.

[33] M. Kaushik, V. Prasad, K. M. Krishna, and B. Ravindran, "Overtaking maneuvers in simulated highway driving using deep reinforcement learning," 2018.

[34] A. Kendall, J. Hawke, D. Janz, P. Mazur, D. Reda, J.-M. Allen, V.-D. Lam, A. Bewley, and A. Shah, "Learning to drive in a day," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019.

[35] S. Kazemkhani, A. Pandya, D. Cornelisse, B. Shacklett, and E. Vinitsky, "Gpudrive: Data-driven, multi-agent driving simulation at 1 million fps," 2025.

[36] S. Hagedorn, M. Hallgarten, M. Stoll, and A. Condurache, "The integration of prediction and planning in deep learning automated driving systems: A review," 2024.

[37] W. Zeng, W. Luo, S. Suo, A. Sadat, B. Yang, S. Casas, and R. Urtasun, "End-to-end interpretable neural motion planner," 2021.

[38] N. Rhinehart, J. He, C. Packer, M. A. Wright, R. McAllister, J. E. Gonzalez, and S. Levine, "Contingencies from observations: Tractable contingency planning with learned behavior models," 2021.

[39] S. Pini, C. S. Perone, A. Ahuja, A. S. R. Ferreira, M. Niendorf, and S. Zagoruyko, "Safe real-world autonomous driving by learning to predict and plan with a mixture of experts," 2022.

[40] W. Zheng, R. Song, X. Guo, C. Zhang, and L. Chen, "Genad: Generative end-to-end autonomous driving," 2024.

[41] Y. Chen, P. Karkus, B. Ivanovic, X. Weng, and M. Pavone, "Tree-structured policy planning with learned behavior models," 2023.

[42] B. Jiang, S. Chen, Q. Xu, B. Liao, J. Chen, H. Zhou, Q. Zhang, W. Liu, C. Huang, and X. Wang, "Vad: Vectorized scene representation for efficient autonomous driving," 2023.

[43] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," 2019.

[44] B. Wilson, W. Qi, T. Agarwal, J. Lambert, J. Singh, S. Khandelwal, B. Pan, R. Kumar, A. Hartnett, J. K. Pontes, D. Ramanan, P. Carr, and J. Hays, "Argoverse 2: Next generation datasets for self-driving perception and forecasting," 2021.

[45] S. Ettinger, S. Cheng, B. Caine, C. Liu, H. Zhao, S. Pradhan, Y. Chai, B. Sapp, C. Qi, Y. Zhou, Z. Yang, A. Chouard, P. Sun, J. Ngiam, V. Vasudevan, A. McCauley, J. Shlens, and D. Anguelov, "Large scale interactive motion forecasting for autonomous driving : The waymo open motion dataset," 2021.

[46] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," 2017.

[47] H. Caesar, J. Kabzan, K. S. Tan, W. K. Fong, E. Wolff, A. Lang, L. Fletcher, O. Beijbom, and S. Omari, "Nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles," 2022.

[48] P. Polack, F. Altché, B. d'Andréa Novel, and A. de La Fortelle, "The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?" 2017.

[49] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi, "The curious case of neural text degeneration," 2020.

[50] K. Renz, K. Chitta, O.-B. Mercea, A. S. Koepke, Z. Akata, and A. Geiger, "Plant: Explainable planning transformers via object-level representations," 2022.