Predictive Driver Model: A Technical Report

Daniel Dauner[1,2] **Marcel Hallgarten**[1,3] **Andreas Geiger**[1,2] **Kashyap Chitta**[1,2]
[1]University of Tübingen [2]Tübingen AI Center [3]Robert Bosch GmbH
https://github.com/autonomousvision/tuplan_garage

1 Introduction

The nuPlan competition comprises three distinct challenges: open-loop, closed-loop with nonreactive agents, and closed-loop with reactive agents. We find that rule-based planners excel in closed-loop scenarios but lag behind learned methods in long-term forecasting in the open-loop evaluation and vice versa. Recognizing that closed-loop performance strongly relies on the initial two seconds of the trajectory, while open-loop scores predominantly depend on accurate endpoint estimation, we merge a rule-based planner with learned ego-forecasting techniques. To this end, we build our `PDM` short-term planning module upon the well-known `IDM` planner [1], incorporating ideas from model predictive control. Moreover, we use `GC-PGP` [2] to accurately estimate the endpoint over the full eight seconds planning horizon. Our combined system integrates both sub-planners by interpolating between the short-term and long-term plans. We find that this effectively combines the strengths of both modules. Our method ranked first on the public nuPlan leaderboard and won the 2023 nuPlan challenge. Given its simplicity, it provides a robust starting point for motion planning research. All code and models are publicly released.

2 Predictive Driver Model

In this section, we describe our proposed hybrid planning model composed of a rule-based short-term planning module and a long-term ego-forecasting component. For the short-term planning crucial for closed-loop performance, we extend the `IDM` planner with concepts from model predictive control. We refer to this model as `PDM-Closed`, i.e., Predictive Driver Model (Closed-Loop). Additionally, we leverage an ego-forecasting module to obtain an accurate prediction of the long-term trajectory, which we refer to as `GC-PGP++`. The overall model architecture is shown in Fig. 1.

2.1 Short-Term Planning

Path Planning. In contrast to `IDM`, `PDM-Closed` utilizes Dijkstra, with the lane length as edge weights, to search for a sequence of lanes along the route and extract their centerline. We found Dijkstra slightly more suitable due to avoiding detours while having no substantial effect on runtime.

Observation & Forecasting. We forecast dynamic agents for the planning horizon of 8s by assuming constant velocity and heading angle. The projected bounding boxes, together with the static obstacles, are stored in occupancy maps. We only consider the nearest 50 vehicles, 25 pedestrians, 10 bicycles, and 50 static objects to the ego agent. Thereby, the planner avoids exploding computation costs when near a large number of entities (e.g., a crowd of pedestrians). Like `IDM`, we add intersections on the route with a red traffic light as stationary objects.

Proposals. We generate proposals by pairing 3 centerline offsets and 5 `IDM` policies at varying target speeds, resulting in 15 proposals. The parameters are summarized in Table 1. We use higher acceleration parameters than standard `IDM` to foster progress. We unroll the proposals for a shorter horizon of 4s to avoid high computation costs in subsequent steps.

Simulation. Trajectories in nuPlan are simulated by iteratively retrieving actions from an LQR controller and propagating the ego vehicle with a kinematic bicycle model [3]. As the controller

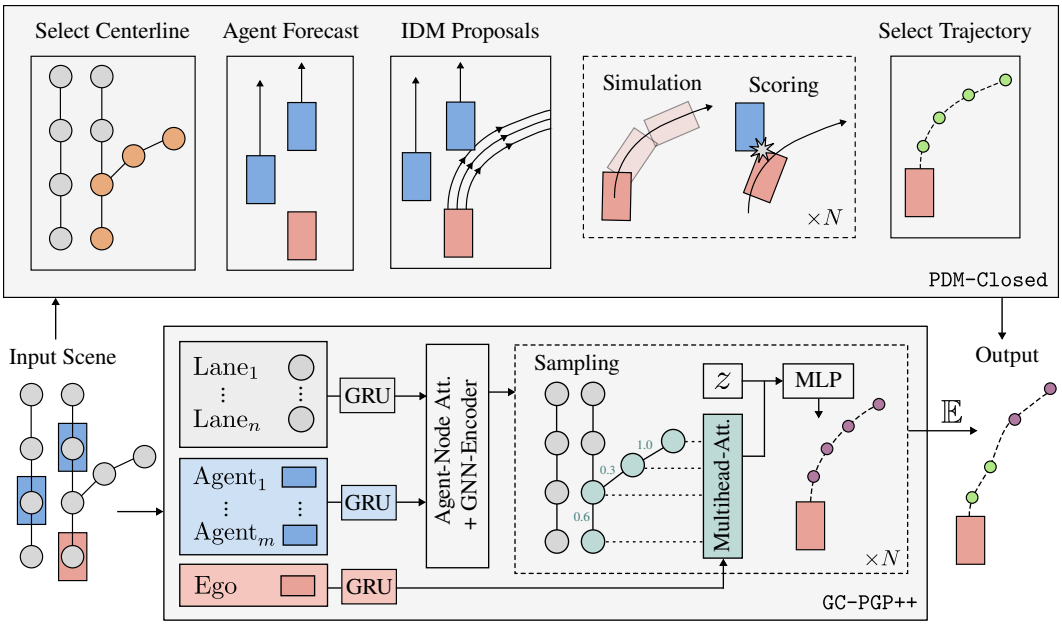

Figure 1: **Architecture.** `PDM-Closed` selects a centerline, forecasts the environment, and creates varying trajectory proposals, which are simulated and scored for trajectory selection. The `GC-PGP++` module aggregates route-constrained trajectory samples via expectation. Our final planner combines both trajectories.

Parameter	Value	Description
o	$\{-1, 0, 1\}$ m	Centerline offsets
v_0	$\{\frac{i}{5} v_{\text{lane}}\}_{i=1,\dots,5}$	Desired velocity. Either the current speed-limit, or $v_{\text{lane}} = 15$ m/s if speed-limit not available.
s_0	1.0 m	Desired net distance to the leading agent $\alpha - 1$.
T	1.5 s	Desired time headway to leading agent $\alpha - 1$.
a	1.5 ms^{-2}	Maximum acceleration of ego vehicle α
b	3.0 ms^{-2}	Maximum deceleration (positive) of ego vehicle α
δ	10.0	Acceleration exponent.

Table 1: `PDM-Closed` Parameters.

is often unable to track the planned trajectory robustly, we simulate the outcome of each proposal before evaluating it.

Scoring. Our scoring function closely resembles the nuPlan closed-loop metrics [4]. 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, driveable area infractions, and driving direction compliance as multiplicative metrics. Furthermore, the scoring evaluates progress, time-to-collision, and comfortability as weighted metrics. We normalize the progress metric with the highest progress of a proposal, which is 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` outputs the highest-scoring proposal which is extended to the complete planning horizon of 8s with the corresponding `IDM` policy. If the best trajectory is expected to collide within 2s, the output is overwritten with an emergency brake maneuver.

2.2 Long-Term Correction

The PDM-Closed component is able to achieve excellent performance in the closed-loop evaluation. However, it is unable to accurately imitate the human expert's long-term plan, resulting in poor open-loop performance. Therefore, our method combines PDM-Closed with an ego-forecasting approach, namely GC-PGP++, which is a modified version of GC-PGP [2]. Finally, we fuse the trajectories of GC-PGP++ and PDM-Closed, resulting in our hybrid planner PDM-Hybrid.

Goal-conditioned Ego-Forecasting via Graph-based Policy. GC-PGP extends the state-of-the-art prediction model, called PGP [5], for goal-directed ego-forecasting.

The model receives an ego-centered lane-graph representation and observed states of surrounding agents and the ego vehicle. The nodes in the lane graph compromise polylines of similar length, with directed edges for lanes in proximity or the direction of traffic flow. The lane nodes and dynamics of the surrounding agents and the ego vehicle are encoded with separate Gated Recurrent Units (GRUs). The model aggregates the information by applying Agent-to-Node Attention and Graph Neural Network (GNN) layers, yielding a per-node feature representation.

The node features are used to estimate transition probabilities for outgoing edges. Subsequently, traversals across the lane graph are sampled. During inference, GC-PGP masks out off-route edges to ensure goal-compliant traversals. Then, a latent-variable model decodes trajectories based on the traversals and the ego-motion encoding. The output trajectories are obtained after a k-means clustering. The original version of GC-PGP selects the cluster centers with the highest rank as output trajectory.

Modifications. We observe that the goal-conditioned trajectories do not describe disjoint behaviors. Hence, we omit the k-means clustering and instead calculate the expectation of all decoded trajectories by averaging. We observed that our modification for GC-PGP++, paired with the hard route constraints of GC-PGP, led to a small OLS performance increase while being computationally more efficient.

2.3 Hybrid Planning System

Our hybrid planning system combines the strong closed-loop performance of PDM-Closed with the open-loop capabilities of GC-PGP++.

Trajectory Fusion. Since the closed-loop performance is almost exclusively determined by the first two seconds of the trajectory, we keep this short-term plan from the PDM-Closed module and append the remaining trajectory from GC-PGP++ to it. This preserves the strong closed-loop performance while only slightly impairing the open-loop score.

Runtime optimization. The planning system is evaluated under stringent time constraints for the online leaderboard evaluation, i.e., with a maximum of 1 second per frame. To save runtime, our planner can distinguish between open- and closed-loop evaluation and infers the rule-based short-term planner only in closed-loop. The detection is done by simulating the trajectory of the previous iteration for one step. If a notable displacement between the simulated and observed position occurs, the planner assumes to be in open-loop evaluation.

Note that in an updated version of our planner (designed immediately following the challenge deadline), we replace the GC-PGP open-loop module with a lightweight and fast multi-layer perceptron. This reduces runtime significantly and simplifies our method. Therefore, we omit the evaluation mode detection in our final version of PDM used in our research paper. All details regarding this updated planner, as well as the insights gained from our preliminary experiments and ablation studies, are available in the full research paper [6]. We released all code and models at https://github.com/autonomousvision/tuplan_garage

Method	Rep.	CLS-R ↑	CLS-NR ↑	OLS ↑	Time ↓
IDM [1]	Centerline	77	76	38	27
PDM-Closed (Ours)	Centerline	**92**	**93**	44	91
GC-PGP [2]	Graph	54	57	82	100
PDM-Open (Ours)	Graph	49	50	**84**	84
Urban Driver [7]	Polygon	44	45	76	64
PlanCNN [8]	Grid	72	73	64	43
PDM-Hybrid (Ours)	Graph	**92**	**93**	**84**	172
Log Replay	*GT*	*80*	*94*	*100*	-

Table 2: **Benchmark Scores.** We show the closed-loop score reactive/non-reactive (CLS-R/CLS-NR), open-loop score (OLS) and runtime in ms. We also mark the input representation (Rep.) used by each planner. PDM-Hybrid accomplishes strong capabilities in both ego-forecasting and planning.

3 Results

Evaluation. We use all 70 scenario types from nuPlan for training and sample a maximum of 4k scenarios per type, totaling ∼178k training scenarios. For evaluation, we use 100 scenarios of the 14 scenario types used for the leaderboard, totaling 1,118 scenarios. Despite minor imbalance (for some scenario types, less than 100 scenarios are available), our validation split aligns well with the online leaderboard results.

Baselines. We include two additional SoTA approaches adopting ego-forecasting for planning in our study. Urban Driver [7] encodes polygons with PointNet layers and predicts trajectories with a linear layer after a multi-head attention block. PlanCNN [8] predicts waypoints using CNN from rasterized grid features without ego state input. It shares several similarities to ChauffeurNet [9], a seminal work in the field.

Training. We train our GC-PGP++ model as proposed in [2] for 70 epochs with a batch size of 32 and a learning-rate of $1e^{-4}$ that is decayed after 40, 50, and 55 epochs by a factor of 0.5.

Hardware Simulation, training, and runtime analysis were performed on an AMD Ryzen 9 7950X CPU, 64GB memory, and a single NVIDIA RTX 3090 GPU.

Results. Our results, presented in Table 2, highlight PDM-Closed's advantages over IDM in terms of CLS. Moreover, PDM-Hybrid successfully combines PDM-Closed's closed-loop excellence with GC-PGP++'s open-loop performance, setting the SoTA for this benchmark. Intriguingly, PlanCNN achieves the best CLS among learned planners, likely due to its design choice of removing the ego state from input, trading OLS for enhanced CLS. Furthermore, the Log Replay planner, outputting the ground-truth ego future trajectory, fails to achieve a perfect CLS, partly due to nuPlan's simplistic LQR controller occasionally drifting from the provided trajectory. PDM-Hybrid compensates for this by evaluating IDM proposals based on the expected controller outcome.

References

[1] M. Treiber, A. Hennecke, and D. Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical review E*, 2000.

[2] M. Hallgarten, M. Stoll, and A. Zell. From Prediction to Planning With Goal Conditioned Lane Graph Traversals. *arXiv.org*, 2302.07753, 2023.

[3] R. Rajamani. *Vehicle dynamics and control*. Springer Science & Business Media, 2011.

[4] Motional. nuplan metrics, 2023. URL https://nuplan-devkit.readthedocs.io/en/latest/metrics_description.html.

[5] N. Deo, E. M. Wolff, and O. Beijbom. Multimodal Trajectory Prediction Conditioned on Lane-Graph Traversals. In *Proc. Conf. on Robot Learning (CoRL)*, 2021.

[6] D. Dauner, M. Hallgarten, A. Geiger, and K. Chitta. Parting with misconceptions about learning-based vehicle motion planning. *Proc. Conf. on Robot Learning (CoRL)*, 2023.

[7] O. Scheel, L. Bergamini, M. Wolczyk, B. Osiński, and P. Ondruska. Urban driver: Learning to drive from real-world demonstrations using policy gradients. In *Proc. Conf. on Robot Learning (CoRL)*, 2021.

[8] K. Renz, K. Chitta, O.-B. Mercea, A. S. Koepke, Z. Akata, and A. Geiger. Plant: Explainable planning transformers via object-level representations. In *Proc. Conf. on Robot Learning (CoRL)*, 2022.

[9] M. Bansal, A. Krizhevsky, and A. S. Ogale. Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. In *Proc. Robotics: Science and Systems (RSS)*, 2019.