Taming an End-to-End Autonomous Driving Policy for Urban Navigation of Quadruped Robots

Joochan Kim[1*•] Chanuk Yang[2*] Tackgeun You[2] Ziran Wang[1] Hwasup Lim[2]

***Abstract*— We present Go2-DrivoR, a goal-conditioned adaptation of the end-to-end autonomous driving trajectory planning framework DrivoR for urban navigation with quadrupedal robots. By conditioning trajectory generation on a local-frame subgoal through a goal token and adapting the vehicle-centric scoring formulation, the method extends DrivoR to short-horizon goal-conditioned local planning without redesigning its core decoders. Specifically, we redefine drivable-area compliance for sidewalk-oriented navigation and reformulate the original ego progress term as goal-conditioned ego progress. Trained exclusively on TartanGround simulation data, Go2-DrivoR improves waypoint-conditioned planning performance on unseen simulation environments and transfers zero-shot to open-loop real-world trajectory prediction.**

I. INTRODUCTION

Urban navigation with quadrupedal robots requires short-horizon goal-conditioned local planning under legged motion constraints. In contrast to autonomous vehicles, quadrupedal robots operate in roadways, walkways, and mixed urban environments where lane centerlines and fixed route geometries are not generally available. A navigation policy must therefore generate local trajectories from visual observations and proprioceptive states while accounting for obstacles, pedestrians, surface variation, and the tracking limitations of a downstream locomotion system. These characteristics make the direct application of vehicle-centric planning methods nontrivial [1], [2].

Recent end-to-end autonomous driving methods integrate learned scene representations with trajectory planning [3], [4]. DrivoR is particularly relevant to this setting because it combines register-token-based visual encoding with a disentangled trajectory-scoring architecture for multi-camera planning [5]. However, DrivoR is designed for vehicle-centric environments, where candidate trajectories are evaluated using road-aligned notions of drivable-area compliance and progress [6], [7]. Although its safety-related terms can be retained with robot-footprint-aware evaluation, the original drivable-area compliance criterion does not encode the relative desirability of sidewalks, roads, and out-of-bounds regions for quadrupedal navigation. Moreover, the original formulation lacks explicit goal conditioning; it neither conditions trajectory proposals on a local lookahead waypoint nor measures progress toward that target.

To address this problem, we adapt DrivoR to short-horizon goal-conditioned quadrupedal navigation while retaining its core decoder architectures. We condition trajectory proposals on a local-frame lookahead waypoint through a goal token. For trajectory scoring, we retain the existing safety criteria with robot-footprint-aware evaluation and redefine drivable-area compliance (DAC) using sidewalk, road, and out-of-bounds regions. We also augment the original ego-progress (EP) term with goal-directed progress to obtain goal-conditioned ego progress (GEP), aligning trajectory evaluation with the supplied local goal.

To assess sim-to-real transfer, we train the model exclusively on simulation data from TartanGround [8]. Real-world logs collected with the Unitree Go2 are withheld from training and used solely for zero-shot open-loop trajectory evaluation. We evaluate the model on seen and unseen simulation environments and real-world Unitree Go2 logs. This protocol evaluates the transferability of the proposed goal-conditioned adaptation without real-world fine-tuning [9].

II. METHODS

A. Go2-DrivoR

We formulate urban quadrupedal navigation as goal-conditioned local trajectory planning. Given four surrounding fisheye images, the robot's proprioceptive state, a high-level directional command, and a two-dimensional goal in the current ego frame, Go2-DrivoR generates K candidate trajectories and selects the one with the highest predicted score. The output consists of eight (x, y, θ) waypoints over a four-second horizon, expressed in the current ego frame.

Our model builds on DrivoR [5], retaining its visual feature extractor, trajectory decoder, scoring decoder, and winner-takes-all trajectory objective. We introduce goal-token conditioning for trajectory generation and adapt the scoring criteria through sidewalk-oriented drivable-area compliance and goal-conditioned ego progress (GEP).

1) Ego-Proprioceptive and Goal Token Extraction: We gather the robot's physical status $\mathbf{s}_{\text{ego}} \in \mathbb{R}^S$ containing proprioceptive signals, including relative ego pose, linear and angular velocities, acceleration, and command inputs. This state is projected by a linear layer to obtain the ego status token $\mathbf{e}_{\text{ego}} \in \mathbb{R}^D$. For each local planning sample, we compute a two-dimensional goal $\mathbf{g} = (x, y) \in \mathbb{R}^2$ by transforming the waypoint four seconds after the current timestamp into the current ego-centric frame. The coordinates are normalized by a distance scale $d_{\text{scale}} = 4\,\text{m}$:

$$\bar{\mathbf{g}} = \left(\frac{x}{d_{\text{scale}}}, \frac{y}{d_{\text{scale}}} \right) \in \mathbb{R}^2. \tag{1}$$

[1]Purdue University
[2]Korea Institute of Science and Technology (KIST)
*Authors have equal contributions
•Work was done when the author was at KIST

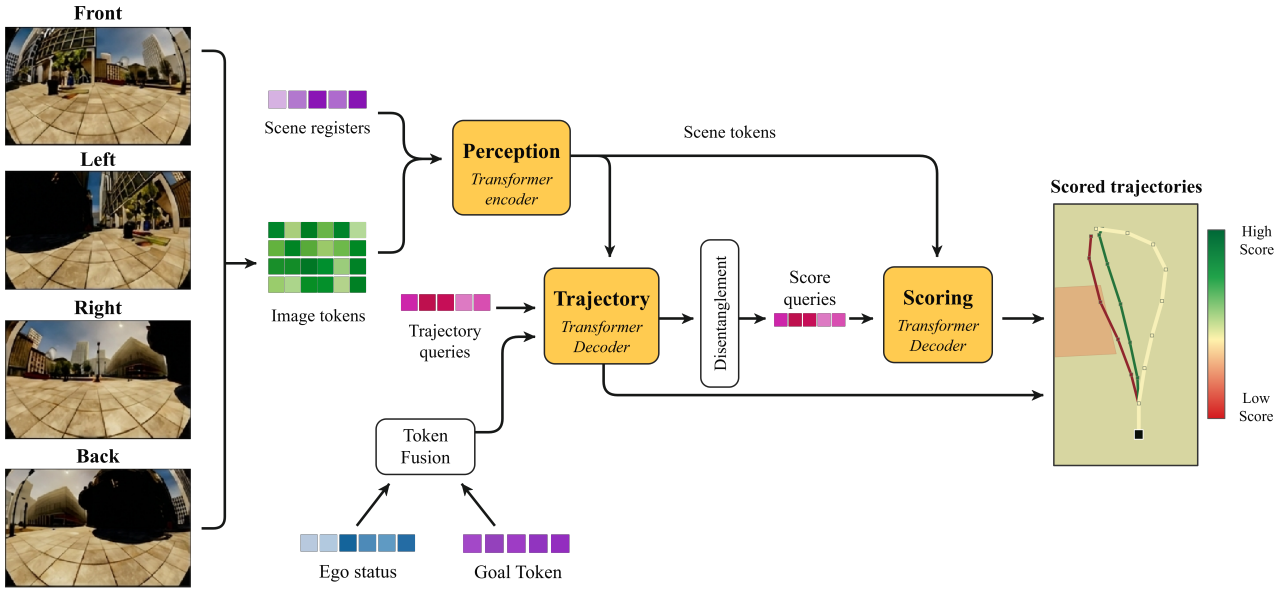

Fig. 1. **Overall framework of Go2-DrivoR.** Multi-camera fisheye images are encoded into scene tokens, while fused ego-state and goal tokens condition trajectory queries. The generated trajectories are ranked using adapted scoring criteria, retaining DrivoR's trajectory and scoring decoder architectures.

The normalized subgoal is projected into the latent space to obtain the goal token:

$$\mathbf{e}_{\text{goal}} = \mathbf{W}_{\text{goal}}\bar{\mathbf{g}} + \mathbf{b}_{\text{goal}}, \tag{2}$$

where $\mathbf{W}_{\text{goal}} \in \mathbb{R}^{D\times 2}$ and $\mathbf{b}_{\text{goal}} \in \mathbb{R}^{D}$ are learnable parameters.

a) Goal Token Fusion and Trajectory Query Conditioning: The ego status token and goal token are concatenated and projected back to the model dimension:

$$\tilde{\mathbf{e}}_{\text{ego}} = \mathbf{W}_{\text{fuse}}[\mathbf{e}_{\text{ego}}; \mathbf{e}_{\text{goal}}] + \mathbf{b}_{\text{fuse}}. \tag{3}$$

We retain the original winner-takes-all trajectory training objective of DrivoR. Each learnable trajectory query is conditioned by residual addition of the fused token:

$$\tilde{\mathbf{q}}_k = \mathbf{q}_k + \tilde{\mathbf{e}}_{\text{ego}}, \quad k \in \{1,\dots,K\}. \tag{4}$$

The conditioned queries are passed to the unchanged trajectory decoder, which generates $K = 64$ candidate trajectories $\tau_k \in \mathbb{R}^{n_p\times 3}$. Following DrivoR, each trajectory comprises $n_p = 8$ future (x, y, θ) waypoints sampled at 0.5-second intervals over a four-second horizon.

2) Trajectory Scoring for Urban Navigation: Compared with the original six-component scoring formulation of DrivoR, our adapted scorer uses five supervised components: no-fault collision (NOC), drivable-area compliance (DAC), time-to-collision (TTC), goal-conditioned ego progress (GEP), and comfort. The driving-direction compliance (DDC) component is omitted because quadrupedal robots are not constrained to follow a prescribed road or lane direction during destination-based navigation. The following paragraphs describe the robot-footprint-aware safety evaluation and the adapted DAC and GEP formulations.

a) Safety and Drivable-Area Criteria: We retain DrivoR's NOC and TTC formulations but replace the vehicle footprint with the quadrupedal robot footprint for collision and time-to-collision evaluation. To reflect sidewalk-oriented navigation, we redefine DAC using semantic regions along each candidate trajectory. A candidate receives a DAC score of 1 if all sampled waypoints lie on the sidewalk, 0.5 if at least one waypoint lies on the road while no waypoint lies outside the allowable region, and 0 otherwise. These semantic-region labels are used to construct the DAC oracle targets for scoring supervision.

b) Goal-Conditioned Ego Progress (GEP): To explicitly account for progress toward a specified local destination, we augment the original ego-progress term with a goal-directed progress (GP) component. Given a goal $\mathbf{g} \in \mathbb{R}^2$ in the current ego frame, this component measures the candidate trajectory's endpoint displacement projected onto the goal direction:

$$\texttt{goal}(\tau_k) = \text{clip}\left(\frac{(\mathbf{p}_k^{(n_p)} - \mathbf{p}_k^{(0)})\cdot\hat{\mathbf{g}}}{\min(\|\mathbf{g}\|, d_{\text{scale}})}, 0, 1\right), \tag{5}$$

where $\hat{\mathbf{g}} = \mathbf{g}/\|\mathbf{g}\|_2$ denotes the unit vector pointing toward the goal, $\mathbf{p}_k^{(0)} = \mathbf{0}$ is the current robot position in the ego frame, and $\mathbf{p}_k^{(t)} \in \mathbb{R}^2$, $t = 1,\dots,n_p$, denotes a predicted future position at future time step t.

The GEP target is defined by combining the original ego-progress target with the goal-directed progress value:

$$\texttt{gep}(\tau_k) = (1-\alpha)\texttt{ep}(\tau_k) + \alpha\texttt{goal}(\tau_k), \tag{6}$$

where $\alpha \in [0, 1]$ and $\alpha = 0.5$ in our experiments unless specified. This formulation retains the original progress signal while additionally rewarding motion toward the local goal.

TABLE I

OPEN-LOOP EVALUATION ON TARTANGROUND. BEST VALUES, EXCLUDING THE GT AGENT, ARE BOLD.

Split	# Maps	# Scenarios	Method	NOC ↑	TTC ↑	DAC ↑	GEP ↑	Comf. ↑	Score ↑
Seen	6	2,554	GT Agent	0.9636	0.9318	0.9954	1.0000	0.9405	0.9304
			Constant Vel.	0.8728	0.9277	0.9543	0.5339	**1.0000**	0.6953
			iPlanner	0.8978	0.9046	0.9838	0.7690	0.5932	0.7344
			Go2-DrivoR	**0.9397**	**0.9391**	**0.9869**	**0.7875**	0.9894	**0.8368**
Unseen	3	2,707	GT Agent	0.9605	0.9522	0.9956	0.9950	0.9102	0.9269
			Constant Vel.	0.8973	0.9495	0.9032	0.5957	**1.0000**	0.7425
			iPlanner	0.9372	0.9400	0.9836	**0.7827**	0.5962	0.7721
			Go2-DrivoR	**0.9579**	**0.9539**	**0.9888**	0.7757	0.9907	**0.8512**

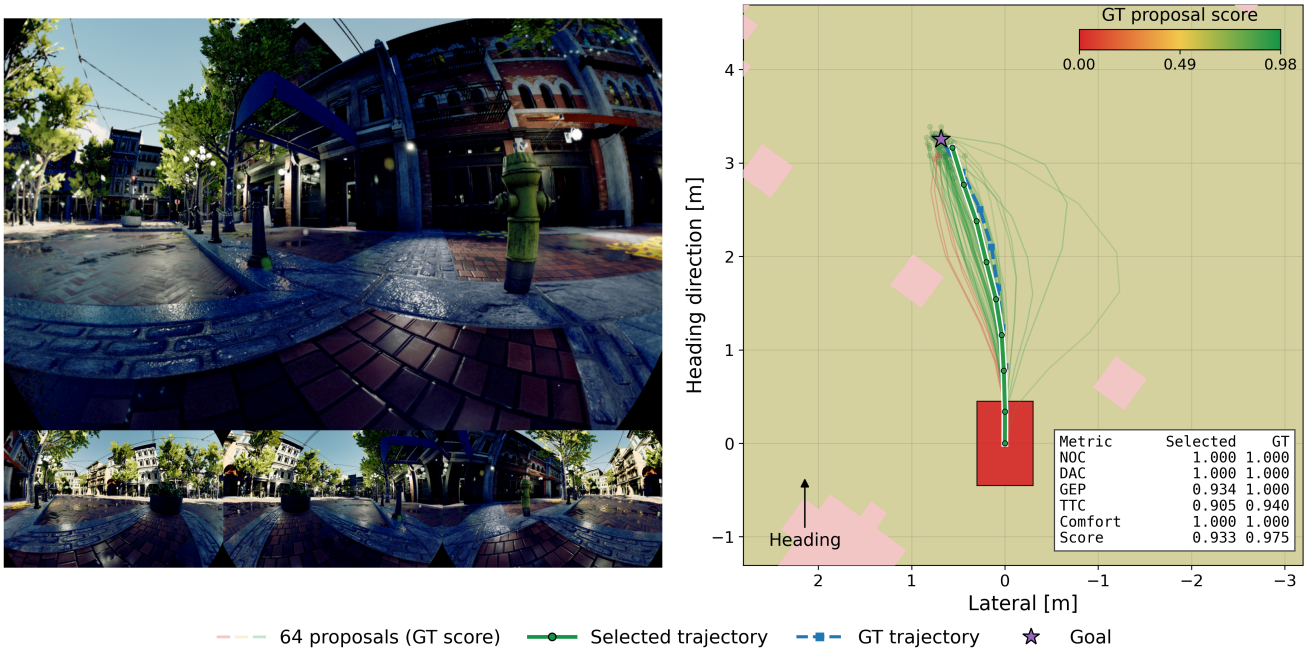

Fig. 2. **Open-loop results on TartanGround.** The left panel shows the input fisheye images: front at the top, and left, rear, and right views from left to right along the bottom. The right panel shows candidate trajectories colored by oracle-evaluated PDMS-style scores, the selected trajectory in green, the expert trajectory in dashed blue, and the goal marked by a star. Yellow denotes walkable ground, and pink denotes non-traversable areas. The red rectangle indicates the robot footprint. The "Selected" and "GT" columns report oracle-evaluated sub-scores and aggregate scores for the selected and expert trajectories, respectively. For the selected trajectory, the scoring decoder predicts NOC $= 0.993$, DAC $= 1.000$, TTC $= 0.994$, GEP $= 0.924$, and comfort $= 1.000$, yielding an aggregate PDMS-style score of 0.959.

c) Composite Ranking Score: At inference, the predicted sub-scores are combined using the log-domain formulation of the original DrivoR scorer. Let $\ell_{k,c}$ denote the predicted logit for candidate trajectory τ_k and score component c, and let

$$\hat{m}_{k,c} = \sigma(\ell_{k,c}) \tag{7}$$

denote the corresponding sigmoid-transformed sub-score. Since DDC is omitted and GEP replaces the original EP term, the composite score is computed as

$$\begin{aligned} s(\tau_k) = {} & \log\left(\hat{m}_{k,\text{noc}}\right) + \log\left(\hat{m}_{k,\text{dac}}\right) \\ & + \log\left(5\hat{m}_{k,\text{ttc}} + 5\hat{m}_{k,\text{gep}} + 2\hat{m}_{k,\text{comfort}}\right). \end{aligned} \tag{8}$$

The composite score is used for proposal ranking and is not directly optimized; the scoring heads are supervised independently using component-wise BCE-with-logits losses.

For evaluation, we compute an adapted PDMS-style aggregate score from oracle-evaluated sub-scores:

$$\begin{aligned} A(\tau) = {} & \frac{\text{NOC}(\tau)\text{DAC}(\tau)}{12} \\ & \times \left(5\,\text{TTC}(\tau) + 5\,\text{GEP}(\tau) + 2\,\text{Comfort}(\tau)\right). \end{aligned} \tag{9}$$

d) Scoring Loss: The scoring decoder is trained using component-wise binary cross-entropy-with-logits losses:

$$\mathcal{L}_{\text{score}} = \sum_{c \in \mathcal{C}} \frac{1}{K} \sum_{k=1}^{K} \text{BCEWithLogits}\left(\ell_{k,c}, m_{k,c}\right), \tag{10}$$

where $\mathcal{C}$ is the set of supervised components, and $m_{k,c}$ and $\ell_{k,c}$ are the oracle target and predicted logit for candidate τ_k and component c.

We jointly optimize the trajectory and scoring losses with equal weights. Both update the shared perception encoder,

TABLE II

ZERO-SHOT PERFORMANCE ON REAL-WORLD DATASET.

Scene	# Scenarios	Method	ADE (m) ↓	FDE@1s (m) ↓	FDE@2s (m) ↓	FDE@4s (m) ↓	Heading MAE (°) ↓
Pedestrian	5990	Constant Vel.	0.785	0.318	0.672	1.470	8.94
		Go2-DrivoR	**0.200**	**0.149**	**0.233**	**0.209**	**5.79**
Road	719	Constant Vel.	0.824	0.333	0.704	1.542	9.22
		Go2-DrivoR	**0.213**	**0.164**	**0.263**	**0.186**	**6.81**

Fig. 3. **Qualitative results on real-world open-loop evaluation.** The expert trajectory, selected trajectory, and constant-velocity prediction are shown in dashed blue, green, and dashed orange, respectively. Purple stars mark the four-second lookahead goals. For the selected trajectory, the scoring decoder predicts NOC = 0.993, DAC = 0.754, TTC = 0.916, GEP = 0.906, and comfort = 1.0, with an aggregate PDMS-style score of 0.693.

but scoring gradients do not pass through predicted trajectories to the trajectory decoder.

B. Simulation and Real-World Data Collection

We train the planner on high-fidelity simulation data and evaluate its zero-shot sim-to-real transfer on real-world Unitree Go2 logs. Both domains undergo a standardized preprocessing and serialization pipeline to unify sensor modalities, coordinate systems, and high-level routing interfaces into the planner's input format.

1) TartanGround: TartanGround [8] serves as our simulation dataset, providing photorealistic multi-camera renderings, IMU telemetry, and dense semantic point clouds across diverse synthetic environments. We process the data through a three-stage pipeline. First, the original multi-camera pinhole images provided by TartanGround are reprojected into four fisheye views—front, left, right, and rear—using a calibrated double-sphere camera model. Second, the resulting images are paired with expert kinematics, and pose, velocity, and body acceleration are transformed from the simulation coordinate system to ENU (East–North–Up) and FLU (Front–Left–Up) frames. To follow the NAVSIM convention, we convert the long continuous TartanGround trajectories into local planning samples using a sliding-window procedure. For a sample at time t, the local-frame goal is defined as the expert pose four seconds after t and is expressed in the current ego frame. This four-second lookahead waypoint is used as the conditioning goal for the corresponding local planning sample, rather than as a global destination. High-level four-way commands are then synthesized from the lateral displacement of the same four-second lookahead waypoint in the ego frame. Finally, semantic point clouds are projected onto a 2D grid to construct boundary, roadway, and walkway vector layers for scoring.

2) Real-world: For real-world data collection, the Unitree Go2 is equipped with four GMSL Owl cameras mounted at 90° azimuthal intervals around the robot, providing front, left, rear, and right views. A human operator teleoperated the robot during data collection, and the recorded robot motion serves as the expert trajectory for open-loop evaluation. The collected logs consist of multi-camera video streams and synchronized state records indexed by frame markers. We align the visual streams with the proprioceptive logs via temporal cross-referencing, and the raw fisheye images are used directly as input without rectification. The kinematic states are converted from the robot's native FRD (Front-Right-Down) frame to FLU, and goal-oriented commands are inferred from lookahead targets using a lateral threshold before serialization.

III. EXPERIMENTS

A. Implementation Details

The training data consist of 14,192 local planning samples from the TartanGround dataset [8]. We train the model for 20 epochs using AdamW [10] on three NVIDIA H200 GPUs with a global batch size of 48. The learning rate is set to 5×10^{-4}, with a two-epoch warm-up phase, followed by cosine annealing. Open-loop inference is performed on a single NVIDIA H200 GPU, with measured batch latencies of 10.8 ms at batch size 1 and 94.1 ms at batch size 16.

B. Open-loop Evaluation

1) Simulator: We compare Go2-DrivoR with a constant-velocity extrapolation baseline and iPlanner [11], and report the logged expert trajectory (GT Agent) as an oracle reference. For iPlanner, we use the official pretrained checkpoint without fine-tuning, providing a front-facing pinhole depth image and the same local goal as Go2-DrivoR. We convert its output path into a four-second trajectory by sampling at 0.5 s intervals using the current ego speed, with headings determined by the path tangent. After reaching the path endpoint, the position is held fixed.

We first evaluate Go2-DrivoR in an open-loop setting on TartanGround [8] to assess trajectory prediction and scoring quality under both seen and unseen maps. Following the

TABLE III

ABLATION STUDY ON GOAL-TOKEN CONDITIONING AND PROGRESS BLENDING.

Variant	Goal Conditioning		Metrics			
	Goal Token	α	ADE (m) ↓	FDE@4s (m) ↓	Goal Progress (GP) ↑	Ego Progress (EP) ↑
EP only	✓	0.0	0.251	0.239	0.979	0.970
w/o Goal Token	×	0.5	4.364	7.766	0.718	0.709
Go2-DrivoR	✓	0.5	**0.202**	**0.207**	**0.988**	**0.976**
Goal-only	✓	1.0	0.325	0.463	0.932	0.901

standard benchmark protocol, we report no-fault collision (NOC), time-to-collision (TTC), drivable-area compliance (DAC), goal-conditioned ego progress (GEP), comfort, and an adapted PDMS-style [6] aggregate score in Eq. (9).

Table I shows that Go2-DrivoR achieves higher aggregate scores than both constant-velocity and zero-shot iPlanner on seen and unseen maps. Compared with iPlanner, Go2-DrivoR achieves higher average NOC, TTC, DAC, and comfort scores on both splits. While iPlanner achieves slightly higher GEP on unseen maps, Go2-DrivoR achieves a higher aggregate score across the evaluated criteria.

The qualitative example in Fig. 2 shows the selected trajectory advancing toward the supplied local goal while remaining within the allowable region. Although it differs from the logged expert path, it achieves an oracle-evaluated PDMS-style score of 0.933, illustrating that a high-scoring trajectory need not exactly reproduce the expert motion.

2) Real-world: We also evaluate zero-shot transfer on real-world logs collected with the Unitree Go2. We evaluated without additional training on the real-world data, and the collected logs are used only for testing. We report average displacement error (`ADE`), final displacement error at multiple horizons (`FDE@1s`, `FDE@2s`, and `FDE@4s`), and heading mean absolute error (`Heading MAE`).

As shown in Table II, Go2-DrivoR outperforms the constant-velocity baseline across all reported metrics and both scene categories. As expected from conditioning on the four-second lookahead waypoint, the largest reduction occurs at FDE@4s; Go2-DrivoR also improves ADE, shorter-horizon FDEs, and Heading MAE. The method produces trajectories that better match the ground-truth future motion and yields improved heading consistency, demonstrating effective zero-shot sim-to-real transfer.

The qualitative result in Fig. 3 further shows that the predicted trajectories are better aligned with the intended navigation direction. In particular, in the intersection example, Go2-DrivoR selects a turning trajectory toward the goal, whereas the constant-velocity baseline continues along the initial heading, illustrating the effect of goal-conditioned trajectory selection. Overall, these results demonstrate zero-shot transfer to goal-conditioned trajectory prediction on real-world logs under the evaluated open-loop protocol.

C. Ablation Studies

We investigate the contributions of explicit goal-token conditioning and progress blending. All variants are trained exclusively on TartanGround and evaluated on the same real-world Unitree Go2 logs described in Section II-B.2, using the same open-loop evaluation protocol. Results are averaged over all scenarios from both the pedestrian and road categories. The goal-token ablation retains $\alpha = 0.5$ in the GEP formulation of Eq. (6), whereas the progress ablations retain goal-token conditioning and vary $\alpha \in \{0, 0.5, 1\}$.

As shown in Table III, removing the goal token while retaining the blended progress objective leads to a substantial degradation in both trajectory prediction accuracy and goal-oriented metrics. A possible explanation is that coarse directional commands leave local paths ambiguous in open urban spaces, whereas an explicit goal provides a precise spatial target. However, it remains unclear why the ablated model exhibits substantially larger trajectory prediction errors than constant-velocity extrapolation despite receiving ego-motion inputs. Further analysis is needed to determine whether the degradation arises primarily from candidate generation or trajectory scoring.

We further investigate the effect of the progress formulation by varying α, which controls the balance between the original ego-progress target and goal-directed progress. Both the EP-only and Goal-only variants perform worse than the blended formulation used in Go2-DrivoR. In particular, relying solely on goal-directed progress does not provide the same trajectory accuracy or progress quality as combining it with the original ego-progress signal. These results support combining the existing ego-progress signal with goal progress in the evaluated setting.

IV. CONCLUSION

We present Go2-DrivoR, a lightweight goal-conditioned adaptation of DrivoR for urban quadrupedal navigation. By conditioning trajectory generation on a local-frame subgoal, redefining drivable-area compliance for sidewalk-oriented navigation, and reformulating ego progress as GEP, the method enables destination-based local planning without redesigning the core decoders. Trained exclusively on TartanGround, Go2-DrivoR improves planning performance on unseen simulation maps and transfers zero-shot to real-world Go2 trajectory prediction. Limitations include evaluation with expert-derived lookahead goals, reliance on open-loop real-world evaluation, and the lack of explicit dynamic-agent modeling. Future work will focus on closed-loop social navigation adaptation, temporal scene modeling, and improved sim-to-real adaptation.

APPENDIX

A. Preliminary: DrivoR

DrivoR [5] is a vision-centric end-to-end autonomous driving framework that maps multi-camera image inputs and ego states to a set of candidate trajectories and their scores. It consists of a perception encoder, a trajectory decoder, and a disentangled scoring decoder.

1) Visual Feature Extraction: DrivoR uses a DINOv2-pretrained ViT-S backbone [12], based on the Vision Transformer (ViT) architecture [13], and fine-tunes it with LoRA [14]. Building on the register-token formulation of Darcet et al. [15], DrivoR introduces camera-specific learnable registers that compress patch-level visual features into a compact scene representation. Each camera is processed independently, and the resulting camera-aware register tokens are collected to form

$$F \in \mathbb{R}^{N_c \times N_r \times D},$$

where N_c is the number of cameras, N_r is the number of registers per camera, and D is the feature dimension.

2) Trajectory Decoder: Given the scene representation F, the trajectory decoder generates K candidate trajectories using learnable trajectory queries $\{q_k\}_{k=1}^{K}$. The original DrivoR formulation embeds ego-state information, including pose, velocity, acceleration, and driving command, and adds this embedding to the trajectory queries before they enter the decoder. Each query then attends to F through cross-attention and is decoded into a trajectory $\tau_k \in \mathbb{R}^{n_p \times 3}$ in the ego frame. During training, the trajectory decoder is optimized using a winner-takes-all (WTA) regression objective, in which only the candidate closest to the expert trajectory is supervised.

3) Scoring Decoder: DrivoR employs a separate scoring decoder to evaluate the generated trajectory candidates. Each decoded trajectory τ_k is detached from the trajectory-generation branch, re-embedded into a scoring query using an MLP, and processed by a transformer decoder that cross-attends to the same scene representation F. This separation prevents scoring gradients from propagating into the trajectory decoder while still allowing the perception encoder to learn features useful for both generation and scoring.

The scoring decoder predicts six component-wise scores derived from the Predictive Driver Model Score (PDMS) [6]: no-fault collision (NOC), drivable-area compliance (DAC), driving-direction compliance (DDC), ego progress (EP), time-to-collision (TTC), and comfort. The individual components are supervised using binary cross-entropy against oracle scores. Although DDC is predicted and supervised, its aggregation weight is zero in the NAVSIM-v1 PDMS formulation used by DrivoR. At inference, the predicted subscores are combined to rank the candidate trajectories, and the highest-scoring candidate is selected.

B. Related Work

1) Robot Navigation: Classical robot navigation systems typically decompose perception, mapping, planning, and control, with representative local-planning approaches including artificial potential fields [16] and the dynamic window approach [17]. Learning-based approaches have increasingly replaced parts of this pipeline with policies or planners that directly predict local paths from perceptual observations and navigation goals.

iPlanner [11] predicts goal-directed local paths from depth observations using imperative learning, while ViPlanner [18] extends this paradigm by incorporating geometric and semantic information for terrain-aware local navigation. Beyond local planning, ViNT [19] learns a general-purpose visual navigation policy from diverse cross-robot navigation data, and NoMaD [20] extends goal-conditioned visual navigation with a diffusion policy and goal masking to unify goal-reaching and exploration.

Language-conditioned embodied agents have also incorporated semantic instructions and pretrained models for navigation and task completion [21], [22]. More recently, navigation foundation models such as NavFoM [23] aim to transfer navigation knowledge across tasks and robot embodiments. In contrast, our work focuses on short-horizon quadrupedal local planning in which multiple goal-conditioned trajectories are explicitly generated and ranked using disentangled criteria for safety, drivability, and progress toward a supplied local goal.

2) End-to-End Autonomous Driving: End-to-end autonomous driving has increasingly adopted architectures that map multi-modal sensor observations directly to planned trajectories. TransFuser [3] integrates visual and LiDAR features for end-to-end perception and planning, while diffusion-based planners model multi-modal future trajectory distributions and enable flexible trajectory generation [24], [25], [26], [27].

SparseDrive [4] demonstrates efficient end-to-end planning using sparse scene representations, whereas DrivoR [5] uses compact camera-aware register tokens together with separate trajectory-generation and trajectory-scoring decoders.

These methods are primarily designed for vehicle-centric navigation. In particular, the PDMS-based scoring formulation adopted by DrivoR uses vehicle-oriented drivable-area and ego-progress criteria that do not explicitly measure advancement toward an arbitrary local destination coordinate. Our work adapts this candidate-generation and scoring paradigm to quadrupedal navigation by conditioning trajectory proposals on a local-frame goal and incorporating goal-directed progress into trajectory evaluation.

3) Evaluation Platforms and Datasets: Urban navigation and autonomous driving policies are commonly evaluated using either non-reactive logged-data protocols or interactive closed-loop simulation. In non-reactive evaluation, the policy output does not affect subsequent sensor observations or environment states. NAVSIM [6], for example, uses a non-reactive simulation over logged driving scenes to compute safety-, comfort-, and progress-oriented metrics for predicted trajectories.

For ground-robot research, TartanGround [8] provides a large-scale multimodal dataset collected in diverse photo-

realistic simulation environments, including multi-camera images, poses, inertial measurements, and semantic information. In this work, we process TartanGround into short-horizon local planning samples and use the resulting data for training and non-reactive open-loop evaluation.

By contrast, closed-loop simulators feed policy outputs back into the environment so that subsequent observations and interactions depend on the agent's behavior. CARLA [28] provides interactive urban simulation for autonomous driving, while MetaUrban [29] targets urban micromobility and supports point- and social-navigation tasks in procedurally generated interactive environments with heterogeneous mobile agents. These platforms provide complementary tools for interactive evaluation, whereas the present work focuses on non-reactive trajectory evaluation and leaves closed-loop quadrupedal navigation for future study.

ACKNOWLEDGMENT

This work was supported by the Korea Institute of Science and Technology (KIST) Institutional Program (Project No. 26E0061).

REFERENCES

[1] J. Lee, M. Bjelonic, A. Reske, L. Wellhausen, T. Miki, and M. Hutter, "Learning robust autonomous navigation and locomotion for wheeled-legged robots," *Science Robotics*, vol. 9, no. 89, Apr. 2024. [Online]. Available: http://dx.doi.org/10.1126/scirobotics.adi9641

[2] K. Zhou, Y. Mu, H. Song, Y. Zeng, P. Wu, H. Gao, and C. Liu, "Ainav: Large language model-based adaptive interactive navigation," 2025. [Online]. Available: https://arxiv.org/abs/2503.22942

[3] K. Chitta, A. Prakash, B. Jaeger, Z. Yu, K. Renz, and A. Geiger, "Transfuser: Imitation with transformer-based sensor fusion for autonomous driving," *Pattern Analysis and Machine Intelligence (PAMI)*, 2023.

[4] W. Sun, X. Lin, Y. Shi, C. Zhang, H. Wu, and S. Zheng, "Sparsedrive: End-to-end autonomous driving via sparse scene representation," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 8795–8801.

[5] E. Kirby, A. Boulch, Y. Xu, Y. Yin, G. Puy, É. Zablocki, A. Bursuc, S. Gidaris, R. Marlet, F. Bartoccioni, *et al.*, "Driving on registers," in *CVPR*, 2026.

[6] D. Dauner, M. Hallgarten, T. Li, X. Weng, Z. Huang, Z. Yang, H. Li, I. Gilitschenski, B. Ivanovic, M. Pavone, A. Geiger, and K. Chitta, "Navsim: Data-driven non-reactive autonomous vehicle simulation and benchmarking," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[7] W. Sun, X. Lin, K. Chen, Z. Pei, X. Li, Y. Shi, and S. Zheng, "Sparsedrivev2: Scoring is all you need for end-to-end autonomous driving," *arXiv preprint arXiv:2603.29163*, 2026.

[8] M. Patel, F. Yang, Y. Qiu, C. Cadena, S. Scherer, M. Hutter, and W. Wang, "Tartanground: A large-scale dataset for ground robot perception and navigation," in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2025, pp. 20 524–20 531.

[9] F. Bjelonic, F. Tischhauser, and M. Hutter, "Towards bridging the gap: Systematic sim-to-real transfer for diverse legged robots," 2025. [Online]. Available: https://arxiv.org/abs/2509.06342

[10] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=Bkg6RiCqY7

[11] F. Yang, C. Wang, C. Cadena, and M. Hutter, "iPlanner: Imperative Path Planning," in *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023.

[12] M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, M. Assran, N. Ballas, W. Galuba, R. Howes, P.-Y. Huang, S.-W. Li, I. Misra, M. Rabbat, V. Sharma, G. Synnaeve, H. Xu, H. Jégou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski, "DINOv2: Learning Robust Visual Features without Supervision," *Transactions on Machine Learning Research*, 2024.

[13] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=YicbFdNTTy

[14] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "Lora: Low-rank adaptation of large language models," in *International Conference on Learning Representations*, 2022.

[15] T. Darcet, M. Oquab, J. Mairal, and P. Bojanowski, "Vision transformers need registers," in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=2dnO3LLiJ1

[16] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, vol. 2, 1985, pp. 500–505.

[17] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.

[18] P. Roth, J. Nubert, F. Yang, M. Mittal, and M. Hutter, "Viplanner: Visual semantic imperative learning for local navigation," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 5243–5249.

[19] D. Shah, A. Sridhar, N. Dashora, K. Stachowicz, K. Black, N. Hirose, and S. Levine, "Vint: A foundation model for visual navigation," *arXiv preprint arXiv:2306.14846*, 2023.

[20] A. Sridhar, D. Shah, C. Glossop, and S. Levine, "NoMaD: Goal Masked Diffusion Policies for Navigation and Exploration," *arXiv pre-print*, 2023. [Online]. Available: https://arxiv.org/abs/2310.07896

[21] A. Suglia, Q. Gao, J. Thomason, G. Thattai, and G. Sukhatme, "Embodied bert: A transformer model for embodied, language-guided visual task completion," *arXiv preprint arXiv:2108.04927*, 2021.

[22] D. Shah, B. Osinski, B. Ichter, and S. Levine, "LM-nav: Robotic navigation with large pre-trained models of language, vision, and action," in *6th Annual Conference on Robot Learning*, 2022. [Online]. Available: https://openreview.net/forum?id=UW5A3SweAH

[23] J. Zhang, A. Li, Y. Qi, M. Li, J. Liu, S. Wang, H. Liu, G. Zhou, Y. Wu, X. Li, *et al.*, "Embodied navigation foundation model," *arXiv preprint arXiv:2509.12129*, 2025.

[24] 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.

[25] B. Liao, S. Chen, H. Yin, B. Jiang, C. Wang, S. Yan, X. Zhang, X. Li, Y. Zhang, Q. Zhang, and X. Wang, "Diffusiondrive: Truncated diffusion model for end-to-end autonomous driving," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2025, Nashville, TN, USA, June 11-15, 2025*. Computer Vision Foundation / IEEE, 2025, pp. 12 037–12 047.

[26] J. Zou, S. Chen, B. Liao, Z. Zheng, Y. Song, L. Zhang, Q. Zhang, W. Liu, and X. Wang, "Diffusiondrivev2: Reinforcement learning-constrained truncated diffusion modeling in end-to-end autonomous driving," *arXiv preprint arXiv:2512.07745*, 2025.

[27] Y. Zheng, R. Liang, K. Zheng, J. Zheng, L. Mao, J. Li, W. Gu, R. Ai, S. Li, X. Zhan, *et al.*, "Diffusion-based planning for autonomous driving with flexible guidance," in *International Conference on Learning Representations*, 2025.

[28] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*. PMLR, 2017, pp. 1–16.

[29] W. Wu, H. He, J. He, Y. Wang, C. Duan, Z. Liu, Q. Li, and B. Zhou, "MetaUrban: An embodied ai simulation platform for urban micromobility," in *International Conference on Learning Representations*, 2025.