2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

LeGo-Drive: Language-enhanced Goal-oriented Closed-Loop End-to-End Autonomous Driving

Pranjal Paul*[1], Anant Garg*[1], Tushar Choudhary[1], Arun Kumar Singh[2], K. Madhava Krishna[1]

Fig. 1: The proposed method, *LeGo-Drive* estimates a goal location queried with a navigation instruction- *"Park near the bus stop on the front-left"* on a single front-facing camera image and coupled it with a differentiable optimizer-based planner that jointly optimizes the trajectory and the goal location. *(Left)* The proposed architecture is shown along with the gradient flow for joint end-to-end training. *(Right-Top)* Goal improvement from initial estimation in Green to improved location in Red. *(Right-Bottom)* Trajectory output in Red leading to the improved goal location, compared with the trajectory generated by baseline in Green.

***Abstract*— Existing Vision-Language Models (VLMs) produce long-term trajectory waypoints or directly control actions based on their perception input and language prompt. However, these VLMs are not explicitly aware of the constraints imposed by the scene or kinematics of the vehicle. As a result, the generated trajectories or control inputs are likely to be unsafe and/or infeasible. In this paper, we introduce *LeGo-Drive*† , which aims to address these issues. Our key idea is to use the VLM to just predict a goal location based on the given language command and perception input, which is then fed to a downstream differentiable trajectory optimizer with learnable components. We train the VLM and the trajectory optimizer in an end-to-end fashion using a loss function that captures the ego-vehicle's ability to reach the predicted goal while satisfying safety and kinematic constraints. The gradients during the back-propagation flow through the optimization layer and make the VLM aware of the planner's capabilities, making more feasible goal predictions. We compare our end-to-end approach with a decoupled framework where the planner is just used at the inference time to drive to the VLM-predicted goal location and report a goal reaching Success Rate of 81%. We demonstrate the versatility of *LeGo-Drive*† across various driving scenarios and navigation commands, highlighting its potential for practical deployment in autonomous vehicles.**

Keywords: Vision-Language Navigation, End-to-End Autonomous Driving, Differentiable Planner

* Co-first Authors
[1] Robotics Research Centre, IIIT Hyderabad, India
[2] University of Tartu, Estonia
† https://reachpranjal.github.io/lego-drive

I. INTRODUCTION

Language-augmented autonomous driving has gained a remarkable surge of interest in recent years. With the emerging use of Vision Language Models (VLMs), existing methods can make informed decisions based on their scene comprehension capability and deliver high-level driving assistance [1, 2, 3]. Broadly, there are two classes of approaches. The first approach pertains to reactive behaviour that maps language commands (like *go-fast*, *slow* etc.) to control action inputs. The second approach maps higher-level navigation instructions (like *changing lanes*, *overtake* etc.) to long-term trajectory prediction [4]. However, employing VLMs in such a manner for either planning or control has one critical drawback- they are primarily trained in an imitation learning setup with some form of expert demonstrations. As a result, these models are not explicitly aware of underlying constraints that are imposed by the scene (e.g. neighbouring agents) or vehicle kinematics. Thus, the generated trajectories or control inputs are likely to be unsafe or kinematically infeasible. Moreover, trajectory prediction subjective to the "logical abilities" of VLMs may lead to infeasible output due to missed detections or false positives.

In this paper, we present a potential solution for making VLM cognizant of the scene constraints while making predictions. We let VLM just predict a goal position instead of a trajectory or low-level control inputs. The VLM prediction is then fed to a downstream planner, which in our case is

a differentiable trajectory optimizer with some learnable parameters. However, the core challenge is to make language-conditioned goal prediction constrained i.e. predicting a goal that is outside the drivable area is undesirable [1]. In other words, VLM prediction should be aware of the capabilities of the downstream planner.

Thus, our key insight is to train the VLM and the parameters of the optimization layer in an end-to-end fashion with a loss function that captures the ego vehicle's ability to reach the predicted target without violating the constraints, encouraging mutual reinforcement. On the other hand, as the training progresses, the VLM learns to predict goals that can be safely reached by the ego-vehicle as shown in Fig 1. The initial goal prediction falls at the curb edge which is clearly unreachable due to the lane boundary constraints. However, after training, the VLM produces goals that are more centred around the parking space.

Our approach also offers other strong advantages besides making VLM more scene-aware. Our results are based on providing the supervision of a very coarse goal region conditioned on the language command, which is easier to obtain compared to the demonstration of the complete driving trajectory. Second, as discussed in [5, 6, 7], goal-directed planning improves the explainability in autonomous driving. Finally, predicting just the goal position would allow the use of smaller and lightweight networks and consequently less data for training, plus faster inference time.

Prior Art: We improve upon the existing literature in two aspects. We demonstrate the impact of our end-to-end training in feasible goal prediction by comparing it against baseline GLC [8], which does not take the downstream planner into account during training. Second, we also compare our approach against post-hoc corrections, wherein ST-P3 [9] (prior contribution in end-to-end motion planning), tries to reach an infeasible goal prediction generated by the baseline network (Refer Table III).

To summarize, our key contributions are:

1) A novel planning-guided end-to-end VLM-based goal-point navigation solution that predicts and improves the desired state by dynamically interacting with the environment and generating a collision-free trajectory.
2) We augment the goal prediction module with a differentiable trajectory optimizer as a layer in an end-to-end setting, allowing us to update the predicted goal constraint by considering the optimizer's feedback, also ensuring that the learned constraint aids the convergence of the optimizer.
3) We conduct extensive closed-loop experiments with different intricate instructions to test the efficacy of the proposed model under different simulation environments with different lighting and weather conditions.

[1]It is possible that the planner handles infeasible goal prediction by stopping short of the predicted position. However, such post-hoc corrections add more burden on the planning and also defeat the explainability objective of our approach.

II. RELATED WORK

A. Visual Grounding

Visual grounding aims to associate a natural language query with the most relevant visual elements or objects in a visual scene. Visual grounding tasks were previously approached as a Referring Image Segmentation (RIS) task [10, 11], integrating linguistic and visual features within a network which GLC [8] and TRSM [12] uses for the task of identifying navigable regions on the drivable areas based on a language command. However, the work is limited to scene understanding and does not include navigation simulations, as trajectory planning relies on precise goal-point location, which they do not address.

B. Interpretability in End-to-End Autonomous Driving

In recent years, End-to-End (E2E) learning-based research has been a prominent focus. The E2E approach is a unified data-driven learning-based driving paradigm, to ensure safe motion planning in contrast to conventional rule-based designs that optimize each task in a disjoint fashion instead of optimizing for a unified target, leading to compounding errors. UniAD [13] is a current state-of-the-art method on nuScenes dataset [14] which uses diverse scene representation to identify vital components within the P3 [15] framework. ST-P3 [9] is a prior art that explores the interpretability of the vision-based E2E system by learning a set of representative spatiotemporal features across the P3 tasks simultaneously and evaluating the trajectory based on the learned costmap. TCP [16] proposes a unified framework to jointly learn the trajectory and control actions by extracting common features from a shared backbone. These approaches are explored in an imitation learning setup which lacks the understanding of constraints imposed by the scene. Due to computational limit, we chose ST-P3 as our baseline for motion planning over UniAD.

C. Planning-oriented Vision-Language Navigation

More recently, LLMs have shown promising results for their multimodal understanding and natural interaction with humans. Existing works [4, 17, 18] use LLMs to reason driving scenes and predict control inputs. However, they are limited to open-loop settings. Current research [1, 2, 19] focuses on adapting to closed-loop solutions but they either directly estimate the control actions or map them to a set of discrete action spaces. These are coarse and are susceptible to perception errors due to their heavy reliance on VLMs knowledge retrieval capabilities that may generate non-smooth motion for intricate cases like parking, highway merge, etc. which require complex combinations of control actions.

III. DATASET

Simulation Setup: The *LeGo-Drive* dataset expands upon existing datasets [8, 20] for a variety of driving scenarios such as lane changes, parking, navigating through intersections, etc. The data collection procedure consists of two stages: 1) synchronous recording of the ego state with camera

sensor data, as well as traffic agents, and 2) parsing and annotating the collected data with navigation directives. We record 4500 training and 1000 validation data points at 10 FPS and to avoid redundancy between consecutive frames, they are filtered at a distance interval of 10m. For each frame, we collect the ego-agent's states, i.e. position, and velocity, ego lane-center with a 50-meter range in both front & rear directions, front RGB camera image, and traffic agent states (position and velocities) utilizing the rule-based expert agent, all in ego-frame. The dataset is diverse across 6 different towns covering a variety of distinct environments representing various driving scenarios with different lane configurations, traffic densities, lighting, and weather conditions. Additionally, the dataset includes a variety of objects commonly observed in outdoor scenes such as bus stops, food stalls, and traffic signals.

Language Command Annotation: Each frame is labelled manually with proper navigation commands corresponding to goal region segmentation masks to cover a range of driving scenarios. We consider 3 different command categories:

1) *Object-centric commands*, which directly refer to an object visible in the current camera frame
2) *Lane Maneuvering commands*, which are instructions specific to actions related to a lane change or adjusting within the lane
3) *Composite commands*, which connect multiple instructions to simulate real driving scenarios

We utilize ChatGPT API to generate different variants with similar semantic meanings. It bears noting that we do not incorporate handling misleading instructions. This capability is imperative in scene reasoning models which may be considered for future expansion; however, it falls outside the scope of our current study.

IV. METHODOLOGY

LeGo-Drive is a framework devised to address the feasibility of coarse estimation of control actions from VLMs, treating it as a short-term goal-reaching problem. This is achieved through the learning of trajectory optimizer parameters together with behavioural inputs by generating and improving a feasible goal aligned with the navigation instruction. As illustrated in Figure 2, the architecture is composed of two major segments:

A. Goal Prediction through Vision-Language Alignment

The Goal Prediction Module takes a front-facing RGB image along with a navigation instruction and predicts a segmentation mask followed by a goal location. It utilizes CLIP [21] text and image encoders to process navigation commands and extract hierarchical visual features from the front-view image. Hierarchical features are known to be beneficial for semantic segmentation; hence, we extract different visual feature $\mathbf{V_i} \in \mathbb{R}^{C_i \times H_i \times W_i}$ where $i \in \{2, 3, 4\}$ after the 2^{nd}, 3^{rd}, and 4^{th} layers of the ResNet backbone. Each $\mathbf{V_i}$ is passed through convolutional blocks ConvBlock_i to bring them into a standard size with equal channel sizes, heights, and widths.

To capture the multi-modal context from the image and text features, we further use a transformer encoder adopted from the DETR [22] architecture. All features $\mathbf{T}$, $\mathbf{V_2}$, $\mathbf{V_3}$, $\mathbf{V_4}$ are flattened, and text features are concatenated with different $\mathbf{V_i}$ individually to get multimodal features $\mathbf{M_i} = \mathbf{V_i} \oplus \mathbf{T}$. The $\mathbf{M_i}$ is then individually passed to the transformer encoder where the multi-headed self-attention layer helps in cross-modality interaction between the different kinds of features to obtain $\mathbf{X_i}$ as the encoder output with the same shape as $\mathbf{M_i}$. We have two decoder heads, one each for the segmentation mask prediction and the goal point prediction task respectively. To predict the segmentation mask, $\mathbf{X_i}$ undergoes further reshape and restructure operations to reshape it into $\mathbb{R}^{C \times H \times W}$, resulting in $\mathbf{Z_i}$. For the segmentation mask prediction, we stack the $\mathbf{Z_i}$ from all layers to shape $\mathbb{R}^{C+C+C \times H \times W}$. Both prediction heads use ASPP decoders from [23]. For segmentation mask prediction, ASPP outputs pass through a convolutional upsampling block that includes bilinear upsampling at specified stages to increase spatial resolution. The output finally undergoes sigmoid activation to produce binary masks. In the goal point prediction decoder, it consists of convolutional layers followed by fully connected layers with the output reshaped to $\mathbb{R}^{2 \times 1}$ representing a pixel location on the image.

First, the segmentation mask prediction head is trained end-to-end with BCE loss between the predicted segmentation mask and the human-annotated ground truth segmentation mask. After a few epochs, the goal point prediction head is trained similarly end-to-end over MSE loss between the predicted goal point and human annotated ground truth goal point.

Handling Composite Commands: To handle complex instructions where the final goal isn't immediately visible, we decompose them into a sequence of atomic commands using few-shot learning with an LLM. We maintain a list of simple actions like lane changes, turns, object references, etc. which the LLM uses to break down complex commands using few-shot learning. Our pipeline then executes these atomic commands iteratively, treating predicted goal points as intermediate waypoints to reach the final destination. This approach enables navigation through multi-step scenarios.

B. Neural Differentiable Planner

Our planner takes the shape of an optimization problem that is embedded with learnable parameters to improve the downstream task of following the goal generated by the VLM. In the following, we first introduce the basic structure of our trajectory optimizer followed by its integration with a network.

Basic Problem Formulation: We assume access to the lane centre-line and use it to construct the Frenet Frame [24]. The trajectory planning is formulated in this frame and has the advantage that the longitudinal and lateral motions of the car are aligned with the X and Y axis of the Frenet

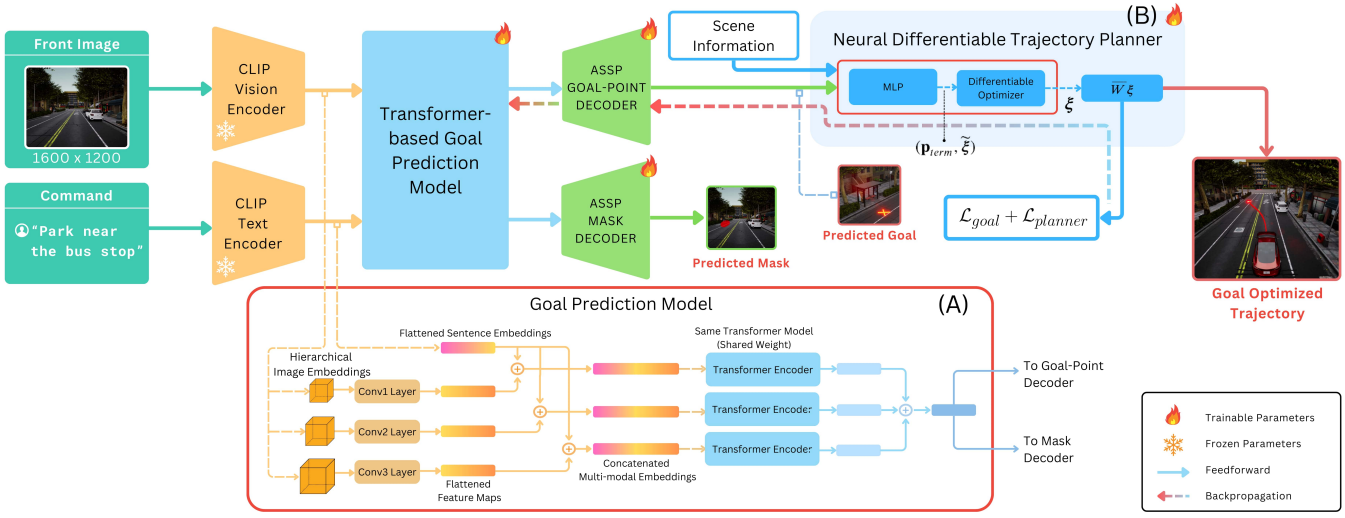

Fig. 2: ***LeGo-Drive* Architecture:** Our architecture comprises of two modules: *(A) Goal Prediction* module and *(B) Differentiable Trajectory Planner*. We propose the advantage of end-to-end training for combined goal and trajectory improvement, for which the gradient-flow is clearly shown. (Refer to Section IV-B for trajectory variable definition)

frame respectively. With this notation in place, our trajectory optimization problem has the following form:

$$\min \sum_{k} \dddot{x}[k]^2 + \dddot{y}[k]^2 \tag{1a}$$

$$(x^{(r)}[0], y^{(r)}[0]) = \mathbf{b}_0, (x^{(r)}[n], y^{(r)}[n]) = \mathbf{b}_f \tag{1b}$$

$$g_i(x^{(r)}[k], y^{(r)}[k]) \leq 0 \tag{1c}$$

where $(x[k], y[k]$ represents the position of the ego-vehicle at time step k. The cost function penalizes high magnitudes of jerk. The equality constraints (1b) ensure that the planned trajectory satisfies the initial and final boundary conditions on the r^{th} derivative of the planned trajectory. We use $r = \{0, 1, 2\}$ in our formulation. The inequality constraints (1c) also depend on the derivatives up to the r^{th} order and include velocity, acceleration, and lane bounds along with constraints on collision avoidance and curvature. The algebraic structures of $g_i(.)$ are taken from our prior work [25]. To ensure that we optimize in the space of smooth trajectories, we parameterize the motions along the $X - Y$ directions in the following form

$$[x[0], x[1], \ldots, x[k]] = \mathbf{W}\mathbf{c}_x, [y[0], y[1], \ldots, y[k]] = \mathbf{W}\mathbf{c}_y, \tag{3}$$

$$\overline{\mathbf{W}} = \begin{bmatrix} \mathbf{W} & \mathbf{0} \\ \mathbf{0} & \mathbf{W} \end{bmatrix} \tag{4}$$

The matrix $\mathbf{W}$ is formed with time dependent cubic-spline basis functions and $(\mathbf{c}_x, \mathbf{c}_y)$ are associated coefficients. Using (4), the optimization (1a)-(1c), can be written in the following compact form:

$$\boldsymbol{\xi}^* = \arg\min_{\boldsymbol{\xi}} \frac{1}{2}\boldsymbol{\xi}^T \mathbf{Q}\boldsymbol{\xi} + \mathbf{q}^T(\mathbf{p})\boldsymbol{\xi}, \tag{5a}$$

$$\mathbf{A}\boldsymbol{\xi} = \mathbf{b}(\mathbf{p}_{term}) \tag{5b}$$

$$\mathbf{g}(\boldsymbol{\xi}) \leq 0 \tag{5c}$$

$$\widetilde{\mathbf{A}}\boldsymbol{\xi} = \widetilde{\boldsymbol{\xi}} \tag{5d}$$

where $\boldsymbol{\xi} = (\mathbf{c}_x, \mathbf{c}_y)$ and $\mathbf{p}_{term}$ is the collection of goal positions and velocities along the x and y component of motion. The matrix $\widetilde{\mathbf{A}}$ and the constant vector $\widetilde{\boldsymbol{\xi}}$ is not part of the original trajectory optimization problem but has been introduced due to some specific reason discussed below.

Conditioning on the Partial Solution: Optimization (5a)-(5c) can be challenging to solve in real-time due the presence of non-convex constraints $\mathbf{g}$. Inspired by [26], we explore a possible solution for accelerating the convergence of the optimizer. Imagine that, we are given a partial solution to the problem. That is, we have some of the components of the solution vector $\boldsymbol{\xi}$ which we denote as $\widetilde{\boldsymbol{\xi}}$ in (5d). We want to use such information to bias the solution process towards favourable regions. To this end, we introduce explicit conditioning through the affine constraints (5d), wherein matrix $\widetilde{\mathbf{A}}$ is simply some specific rows of an identity matrix. More precisely, imagine that $\widetilde{\boldsymbol{\xi}}$ is formed by first ten elements of $\boldsymbol{\xi}$, then $\widetilde{\mathbf{A}}$ will be formed by extracting the first ten rows of the identity matrix of size equal to dimension of $\boldsymbol{\xi}$.

Augmenting Neural Networks : We want to learn the goal states $\mathbf{p}_{term}$ and $\widetilde{\boldsymbol{\xi}}$ in end-to-end fashion along with our VLM model. To this end, we present a hybrid planning network that consists of a multi-layer perceptron (MLP) and differentiable optimization layer embedded with (5a)-(5d). (Refer Figure 2-B). The MLP takes in the embedding from the VLM along with the other scene information such as position and velocities of the neighbouring vehicles to predict $(\mathbf{p}_{term}, \widetilde{\boldsymbol{\xi}})$

C. End-to-End Training Framework:

Due to the modular nature of the architecture, the network architecture of Fig.2 can be trained in two ways:

1) ***LeGo-Drive E2E*:** denotes the joint training of VLM and differentiable planner modules. The model is trained

over the combined loss $\mathcal{L} = \mathcal{L}_{goal} + \mathcal{L}_{planner}$ where goal loss $\mathcal{L}_{goal}$ is the MSE loss calculated between the predicted goal (x_g, y_g) and the endpoint of the planner produced trajectory $(\mathbf{x_{-1}}, \mathbf{y_{-1}})$. The planner loss $\mathcal{L}_{planner}$ is a combination of violation of non-convex constraints $\mathbf{g}$ pertaining to lane offset, collision avoidance and kinematic constraint. The gradient flows from the planner to the goal prediction part as shown in Figure 2.

$$\mathcal{L}_{goal} = \frac{1}{N}\sum_{i=1}^{N} \|(\mathbf{x}_{-1},\ \mathbf{y}_{-1}) - (x_g,\ y_g)\,\|_2^2 \tag{6a}$$

$$\mathcal{L}_{planner} = ||\mathbf{max}\,(0,\ \mathbf{g})\,||_2 \tag{6b}$$

2) ***LeGo-Drive Decoupled*:** denotes the training process where both the goal prediction module and planner module are trained separately. First, the goal prediction module is trained over MSE loss between the ground truth mask centroid and the predicted goal. Subsequently, the planner is trained on $\mathcal{L}_{planner}$ while keeping the parameters of the goal prediction module frozen.

The end-to-end training requires backpropagating through the differentiable trajectory optimization layer , which can be done in two ways namely: implicit differentiation and algorithmic unrolling [27]. The support for the former in terms of existing libraries is mostly restricted to convex problems [28], or unconstrained non-linear least squares [27]. In our approach, we build a custom backprogation routine using algorithm unrolling and following our prior work [25]. An advantage of our approach is that it can handle constraints, and the backpropagation can be made free of matrix factorization [25]. The performance of both methods is shown in Table III which is further analyzed in the subsequent section.

V. EXPERIMENTS AND RESULTS

A. Implementation Details

Our model takes an RGB image of resolution 1600×1200 pixels along with a language instruction (max 20 tokens) as inputs. We use CLIP variants to extract vision and text embeddings. The model predicts a goal location in pixel space which is projected into 3D using the known camera height. This 3D goal location along with surrounding agents' information from the simulator is transformed into the ego-centric frame for planning. The planner operates in the road-aligned Frenet frame, using the ego lane centre as a reference path and adhering to lane and vehicle constraints based on the simulator settings. The planning horizon is 6 timesteps with a step length of 0.5 meters, considering the 5 nearest obstacles within 50 meters radius. We train the model using Adam optimizer with $6e^{-5}$ learning and polynomial learning rate decay of 0.2 for 100 epochs on a single Nvidia RTX 4090 GPU, which takes about 6 hours for end-to-end training.

B. Evaluation Metrics

We evaluate *LeGo-Drive* based on the command type with L2 norm as the primary metric. To consider improvement in goal location, we address: 1). the closeness of the predicted goal w.r.t. to the mask centroid and 2). the lane centre. We also report the proximity to the nearest obstacles agnostic to the command type.

To assess improvement in the trajectory, we evaluate the minFDE (minimum Final Displacement Error) metric adopted from [13], defined as the L2 distance between the goal location and the trajectory endpoint. Further, we assess goal reachability by calculating the Success Rate, defined as the percentage of times the ego-vehicle reaches within 3 meters of the predicted goal location. Also, we gauge Smoothness, by analyzing the trajectory's convergence rate towards the goal, with slower convergence indicating smoother behaviour.

C. Experimental Results

*1) **Goal Improvement using Differentiable Planner**:* Table I compares the goal evaluation metrics between *LeGo-Drive Decoupled* (Initial) and the proposed *LeGo-Drive E2E* (Improved) for different command types. The E2E approach consistently excels on all the metrics. The model closely approximates the mask centroid which is an ideal location in most scenes evident from the qualitative result. Moreover, it performs comparative w.r.t. the obstacle proximity averaging over variety of driving cases, including parking and obstacle avoidance during lane change. However, with $25\% - 30\%$ improvement of the goal location to the lane centre for a single maneuver command, interestingly, the model shows 75% improvement in compound commands which proves the effectiveness of the proposed method. This can be justified by the controlled actions due to the intermediate improved goal corresponding to the first atomic command which compounded to the enhanced performance.

Type	Obstacle ↑		Mask Centroid ↓		Lane Center ↓	
	Dec.	E2E	Dec.	E2E	Dec.	E2E
I	5.5141	**6.5067**	2.7641	**1.2633**	2.1443	**1.8104**
II	4.7855	**5.4258**	3.9494	**2.0360**	3.2525	**2.9559**
III	4.5602	**5.9259**	3.8007	**1.4245**	1.5358	**1.1823**

TABLE I: **Goal Improvement:** Avg. Distance (in meters) of goal prediction w.r.t. obstacle, mask centroid and lane center according to the different command types. *Dec.* refers to the prediction from the *LeGo-Drive Decoupled* and *E2E* refers to the proposed end-to-end version.

Figure 3 shows the qualitative results across various categories of language commands. The visualizations reveal instances where the goal prediction by *LeGo-Drive Decoupled* tends to fall into the infeasible areas. However, through our approach, these goals are subsequently refined and improved. For eg., in Figure 3-(b), the initial goal point falls on top of the car (Green), which transitions to a more viable location away from it (Red). Similarly, improved goals in other results

are closer to the lane centre (a, d) or away from the curb edge (c) facilitating goal-reachability without any post-hoc operation in the trajectory rollout.

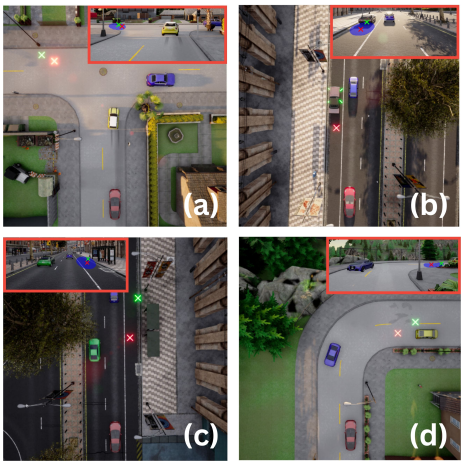

Fig. 3: **Goal Improvement** results for the following commands: (a). *"Take a left turn at the intersection"*, (b). *"Park behind the car on left"*, (c). *"Stop near the bus stop"*, (d). *"Take a right turn ahead"*. The goal locations predicted by the *Decoupled* version (shown in Green) improve to a feasible location (shown in Red) due to differentiable planner augmentation to the perception model and training them end-to-end. They are feasible in the sense that, the predicted locations are closer to the lane centre or are easily reachable without any post-hoc corrections.

*2) **Trajectory Improvement Evaluation**:* We benchmark the planner performance against ST-P3 [9], prior art in an end-to-end motion planner domain. For even comparison, we use the improved goal(s) predicted by the trained *LeGo-Drive E2E* model as the desired location. Table II reports the results for different commands. The proposed approach, benefiting from both goal and trajectory improvement, surpasses the baseline by a large extent. It clearly ensures goal reachability with a high success rate and smooth collision-free trajectories, spanning across commands based on different driving scenarios. There is a significant decrease in minFDE by 60% for composite command which stems from the basic ideology of the proposed model.

Type	minFDE (m) ↓		Smoothness ↓		SR (%) ↑	
	ST-P3	*Ours*	ST-P3	*Ours*	ST-P3	*Ours*
I	0.5145	**0.4639**	0.1836	**0.0384**	50.2	**79.1**
II	0.7710	**0.2985**	0.1679	**0.1365**	36.2	**82.4**
III	0.6477	**0.3982**	0.1836	**0.0603**	45.4	**80.1**

TABLE II: **Trajectory Evaluation:** Evaluating trajectories based on goal reaching. Metrics are defined in Section V-B

Figure 4 compares trajectories generated by our proposed end-to-end approach and the ST-P3 model across various command categories. The visual comparison highlights that, in contrast to the proposed approach, trajectories from ST-P3 frequently deviate from the intended goal location, often leading to misalignment or incorrect orientations.

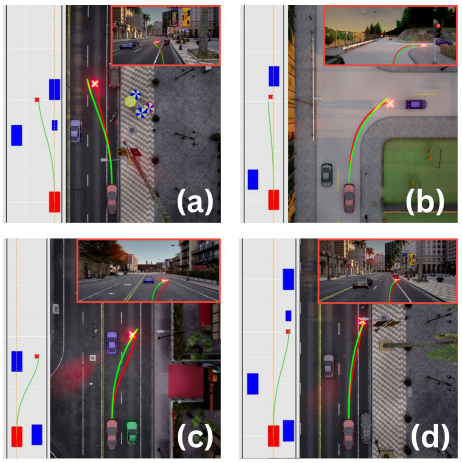

Fig. 4: Qualitative Result of the **Trajectory Improvement** for different commands leading to an *improved* goal. The baseline ST-P3 [9] trajectory (shown in Green) consistently plans a non-smooth trajectory compared to *Ours*, (shown in Red). Each figure also shows our trajectory planning in the Frenet Frame.

Model Comparison: We further compare different training strategies by evaluating trajectory improvement metrics. Based on III, our proposed architecture outperforms the baseline by a percentage difference of 35% in the goal reachability (SR) maintaining smoother convergence. The decoupled version of our model performs relatively well for trajectory improvement with a reasonable difference in Success Rate. However, as previously analyzed in I, the E2E approach shows superior performance in the goal improvement metrics.

Model	minFDE (m) ↓	Smoothness ↓	SR (%) ↑
ST-P3 + GLC	0.5145	0.1836	47.1
Dec. (Ours)	0.3982	0.1662	70.1
E2E (Ours)	**0.2985**	**0.0603**	**81.2**

TABLE III: **Model Comparison:** We compare our proposed approach ***LeGo-Drive E2E*** against the modular decoupled architecture ***LeGo-Drive Dec.*** along with architecture formed by combining baselines [8, 9]. SR refers to Success Rate (in %age)

VI. CONCLUSION

Our study reveals a distinct advantage of the proposed end-to-end approach compared to traditional decoupled methods by solving it as a goal-point navigation problem. The joint training of the goal-prediction module with a differentiable optimizer-based trajectory planner highlights the efficacy of our method leading to enhanced accuracy and context-aware goal forecasting, ultimately resulting in smoother, collision-free navigable trajectories.

VII. ACKNOWLEDGEMENT

The authors acknowledge the support provided by MeitY, Govt. of India, under the project *"Capacity building for human resource development in Unmanned Aircraft System (Drone and related Technology)"*.

The work of Arun Kumar Singh was funded by the European Social Fund and Estonian Research Council via project TEM-TA101, Grant PSG753, and in part by collaboration project LLTAT21278 with Bolt Technologies

REFERENCES

[1] Hao Shao et al. "Lmdrive: Closed-loop end-to-end driving with large language models". In: *arXiv preprint arXiv:2312.07488* (2023) (cit. on pp. 1, 2).

[2] Chonghao Sima et al. "Drivelm: Driving with graph visual question answering". In: *arXiv preprint arXiv:2312.14150* (2023) (cit. on pp. 1, 2).

[3] Hao Tan and Mohit Bansal. *LXMERT: Learning Cross-Modality Encoder Representations from Transformers*. 2019. eprint: 1908.07490 (cs.CL) (cit. on p. 1).

[4] Jiageng Mao et al. "Gpt-driver: Learning to drive with gpt". In: *arXiv preprint arXiv:2310.01415* (2023) (cit. on pp. 1, 2).

[5] Stefano V. Albrecht et al. *Interpretable Goal-based Prediction and Planning for Autonomous Driving*. 2021. eprint: 2002.02277 (cs.RO) (cit. on p. 2).

[6] Amina Ghoul et al. *Interpretable Goal-Based model for Vehicle Trajectory Prediction in Interactive Scenarios*. 2023. eprint: 2308.04312 (cs.AI) (cit. on p. 2).

[7] Junru Gu, Chen Sun, and Hang Zhao. *DenseTNT: End-to-end Trajectory Prediction from Dense Goal Sets*. 2021. eprint: 2108.09640 (cs.CV) (cit. on p. 2).

[8] Nivedita Rufus et al. "Grounding Linguistic Commands to Navigable Regions". In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Sept. 2021 (cit. on pp. 2, 6).

[9] Shengchao Hu et al. "ST-P3: End-to-end Vision-based Autonomous Driving via Spatial-Temporal Feature Learning". In: *European Conference on Computer Vision (ECCV)*. 2022 (cit. on pp. 2, 6).

[10] Yue Liao et al. *A Real-Time Cross-modality Correlation Filtering Method for Referring Expression Comprehension*. 2020 (cit. on p. 2).

[11] Zhengyuan Yang et al. *Improving One-stage Visual Grounding by Recursive Sub-query Construction*. 2020 (cit. on p. 2).

[12] Naoki Hosomi et al. "Trimodal Navigable Region Segmentation Model: Grounding Navigation Instructions in Urban Areas". In: *IEEE Robotics and Automation Letters* 9.5 (2024), pp. 4162–4169 (cit. on p. 2).

[13] Yihan Hu et al. "Planning-oriented Autonomous Driving". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023 (cit. on pp. 2, 5).

[14] Holger Caesar et al. "nuscenes: A multimodal dataset for autonomous driving". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 11621–11631 (cit. on p. 2).

[15] Abbas Sadat et al. *Perceive, Predict, and Plan: Safe Motion Planning Through Interpretable Semantic Representations*. 2020. arXiv: 2008.05930 [cs.RO] (cit. on p. 2).

[16] Penghao Wu et al. "Trajectory-guided control prediction for end-to-end autonomous driving: A simple yet strong baseline". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 6119–6132 (cit. on p. 2).

[17] Hao Sha et al. "Languagempc: Large language models as decision makers for autonomous driving". In: *arXiv preprint arXiv:2310.03026* (2023) (cit. on p. 2).

[18] Zhenhua Xu et al. "Drivegpt4: Interpretable end-to-end autonomous driving via large language model". In: *arXiv preprint arXiv:2310.01412* (2023) (cit. on p. 2).

[19] Long Chen et al. "Driving with llms: Fusing object-level vector modality for explainable autonomous driving". In: *arXiv preprint arXiv:2310.01957* (2023) (cit. on p. 2).

[20] Thierry Deruyttere et al. "Talk2Car: Taking Control of Your Self-Driving Car". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, 2019 (cit. on p. 2).

[21] Alec Radford et al. *Learning Transferable Visual Models From Natural Language Supervision*. 2021 (cit. on p. 3).

[22] Nicolas Carion et al. *End-to-End Object Detection with Transformers*. 2020 (cit. on p. 3).

[23] Liang-Chieh Chen et al. *Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation*. 2018 (cit. on p. 3).

[24] Moritz Werling et al. "Optimal trajectory generation for dynamic street scenarios in a frenet frame". In: *2010 IEEE international conference on robotics and automation*. IEEE. 2010, pp. 987–993 (cit. on p. 3).

[25] Jatan Shrestha et al. "End-to-End Learning of Behavioural Inputs for Autonomous Driving in Dense Traffic". In: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2023 (cit. on pp. 4, 5).

[26] Priya L Donti, David Rolnick, and J Zico Kolter. "DC3: A learning method for optimization with hard constraints". In: *arXiv preprint arXiv:2104.12225* (2021) (cit. on p. 4).

[27] Luis Pineda et al. "Theseus: A library for differentiable nonlinear optimization". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 3801–3818 (cit. on p. 5).

[28] Akshay Agrawal et al. "Differentiable convex optimization layers". In: *Advances in neural information processing systems* 32 (2019) (cit. on p. 5).