VERDI: VLM-Embedded Reasoning for Autonomous Driving

Bowen Feng*†, Zhiting Mei*†, Baiang Li†, Julian Ost†, Roger Girgis‡, Anirudha Majumdar†, Felix Heide†‡
*Equal Contribution †Princeton University ‡Torc Robotics
VERDI-DRIVING.GITHUB.IO

***Abstract*—While autonomous driving (AD) stacks struggle with decision making under partial observability and real-world complexity, human drivers are capable of commonsense reasoning to make near-optimal decisions with limited information. Recent work has attempted to leverage finetuned Vision-Language Models (VLMs) for trajectory planning at inference time to emulate human behavior. Despite their success in benchmark evaluations, these methods are often impractical to deploy (a 70B parameter VLM inference at merely 8 tokens per second requires more than 160G of memory), and their monolithic network structure prohibits safety decomposition. To bridge this gap, we propose VLM-Embedded Reasoning for autonomous DrIving (VERDI), a training-time framework that distills the reasoning process and commonsense knowledge of VLMs into the AD stack. VERDI augments modular differentiable end-to-end (e2e) AD models by aligning intermediate module outputs at the perception, prediction, and planning stages with text features explaining the driving reasoning process produced by VLMs. By encouraging alignment in latent space, VERDI enables the modular AD stack to internalize structured reasoning, without incurring the inference-time costs of large VLMs. We demonstrate the effectiveness of our method on the NuScenes dataset and find that VERDI outperforms existing e2e methods that do not embed reasoning by 10% in ℓ_2 distance, while maintaining high inference speed.**

I. Introduction

Real-world autonomous vehicle deployments have drastically increased in the last three years [1]–[7]. However, a major bottleneck to achieving safe and reliable driving arises from the complexity of real-world environments. Dynamic agents, including pedestrians, cyclists, and human-driven vehicles, along with their interactions, present significant challenges for autonomous vehicles to perceive, predict, and plan. Decision-making under semantic ambiguity, partial observability, and unforeseen so-called edge case scenarios remain a critical challenge for autonomous systems [8].

Highly performant autonomous driving (AD) stacks rely on modularized differentiable end-to-end (e2e) architectures [9]–[12]. Although existing methods are successful in benchmark evaluations, they are often limited to the training datasets consisting of human driving trajectories that lack access to the underlying reasoning processes of the driver [13], [14]. As such, conventional supervised e2e models lack semantic understanding of the driving process, which hinders their real-world applications in unseen scenarios [15]. On the other hand, human drivers respond to challenging decisions and navigate through ambiguities by interpreting visual context and reasoning. In addition, they leverage commonsense knowledge and prior experience to make satisfactory decisions even with incomplete information [16], [17]. This motivates the research question: *How can modular autonomous driving models leverage human-like decision-making processes with reasoning and commonsense knowledge when confronted with real-time challenging driving scenarios on the road?*

To enable reasoning for AD, a growing body of work investigates Vision-Language Models (VLMs) in driving tasks [18]–[24]. Specifically, researchers have augmented AD datasets with VLM annotations to incorporate scene descriptions and explanations [23]–[25], questions and answers [20], [22], and driving rationales [15], [19], [23]. The augmented datasets are then used to train or fine-tune VLMs to be multi-modal foundation models that are capable of generating future behaviors for autonomous vehicles, along with natural language explanations for interpretability. However, these works query fine-tuned multi-modal VLMs during inference time, which could take up to a few seconds. For example, on an OrinX chip, a 4B Qwen model can decode a 1,078-token prompt at 44.5 tokens per second [25]. This latency renders them impractical for real-world [15], safety-critical scenarios, or necessitates circumventions such as hierarchical planning and control, using small VLMs, or optimizing the model [25]. Further, many existing VLM methods forgo multi-modal perception and prediction modules and directly output actions from input queries, making safety decomposition a challenge and allows for more VLM hallucinations at inference time, which we illustrate more in Appendix A.

We introduce VLM-Embedded Reasoning for autonomous DrIving (VERDI) as an approach for distilling latent reasoning capabilities from VLMs. Figure 1 provides an overview of VERDI. During training, the method operates on two parallel models for a given trajectory: (1) a differentiable e2e driving model that processes current ego state and corresponding sensor inputs (multi-view images) to generate future trajectories, and (2) a latent reasoning pipeline that extracts the reasoning process of the ground-truth future trajectory by querying a VLM. The driving pipeline sequentially handles perception, prediction, and planning subtasks through the differentiable modules. In this work, we build upon VAD [10] to create our e2e driving pipeline. The reasoning model prompts the VLM using chain-of-thought [26], in the same subtask sequence. To distill the reasoning capabilities and commonsense knowledge from VLMs to the driving model, VERDI performs latent feature alignment between the driving and reasoning pipelines. This alignment process aims to match the driving model

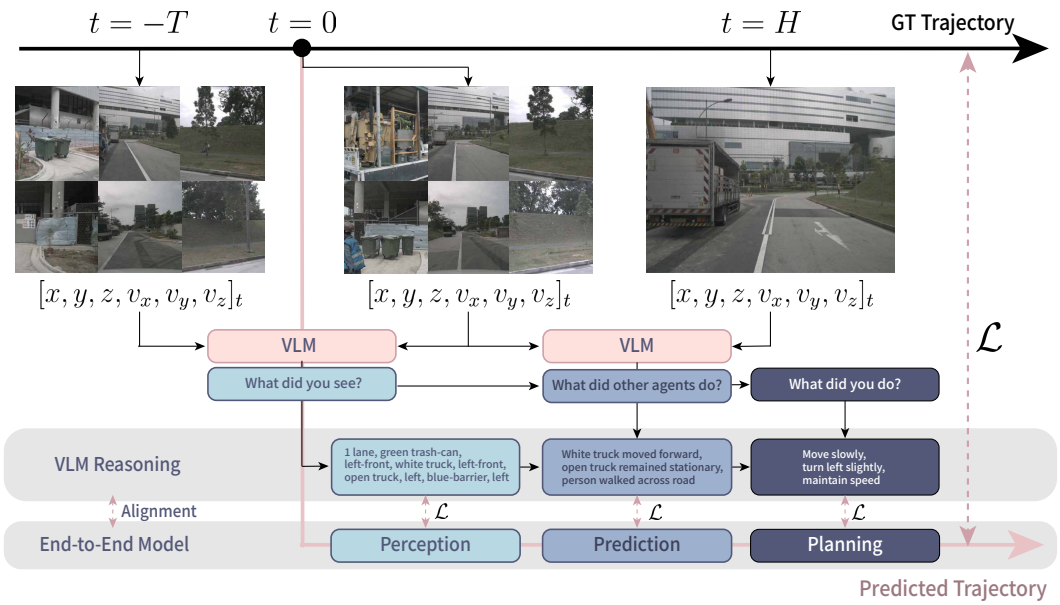


Fig. 1: **Overview of VERDI.** Our pipeline aligns the VLM reasoning module with the e2e driving model. During training, the ground truth (GT) trajectory and observed images are provided to the VLM for it to explain the reasoning throughout perception, prediction, and planning during the driving process. The VLM answers to each submodule is aligned with the corresponding submodule outputs from the e2e driving model, effectively distilling VLM knowledge and reasoning into the e2e model. During inference time, the e2e model plans future trajectory with embedded reasoning process, without having to query the VLM (pink arrow).

latent embeddings with the embeddings of the reasoning process, facilitating the transfer of semantic reasoning from large VLMs to real-time driving model. During inference, the reasoning pipeline is no longer required, since it has been effectively distilled into the driving pipeline. This enables VERDI to perform real-time inference while preserving human-like reasoning abilities. Our main contributions are three-fold: (1) we develop a modular, differentiable, end-to-end autonomous driving (AD) model featuring latent cognitive planning, (2) we introduce a method to align individual driving subtasks with corresponding natural language in the latent space, and (3) we evaluate our framework on the nuScenes dataset [13] and see 10% improvement in ℓ_2 distance over baseline methods; further, we confirm that our model has successfully distilled VLM reasoning processes in each submodule, and elicits qualitative improvements in driving behavior.

II. Related Work

Modularized End to End Autonomous Driving. Although autonomous driving stacks were built modular for a long time, end-to-end optimization is a novel problem [27], [28]. These emerging approaches minimize information loss incurred in independently optimized modules, where errors accumulate and compound throughout the perception, prediction, and planning processes. However, attempts to condense the driving task into single monolithic perception to planning end-to-end models lack the level of interpretability and generalizability desired by AD applications, leading to the rise of jointly trainable end-to-end frameworks with intermediate outputs [10], [11], [29], [30]. Driving innovations arise from advances in perception [31], [32], prediction [33], [34], and planning [35], [36] modules, while differentiable losses spanning the entire pipeline, exemplified by UniAD [9] and SY-P3 [37], serve to connect and unify these components. Improvements in the representation space covering vectorized encoding strategies in VAD [10], [11], splitting dynamic and static environments [38], parallelization [39] and inter modular dependencies as the ego-status [12] have further enhanced performance. However, these models primarily rely on supervised learning from human driving demonstration datasets [13], lacking human reasoning capabilities significantly restricts their effectiveness in challenging driving scenarios.

Vision Language Models (VLMs) in Autonomous Driving are of great interest as they provide advanced reasoning capabilities, and rich semantic scene understandings. Specifically, methods such as DriveGPT4 [21], OmniDrive [22], Driving with LLM [20], DriveMLM [23], [24] enrich existing datasets with fine-grained annotations and fine-tune or train VLMs to reason about the scene through textual responses and predict higher-order actions. EMMA [15] pushes this further by a full integration of perception, road graphs, and control trajectories in an end-to-end fashion, achieving state-of-the art performance. However, large VLMs cause slow inference and deployment

hurdles, forcing compromises, e.g. restricting them to low-frequency planning while an end-to-end model handles high-frequency controls or capping the size below 4B parameters [25], which degrades overall performance. Recent approach VLP [40] uses VLMs during training time only focusing on contrastive alignment between perception representations and language embeddings. In contrast, VERDI exploits the full language reasoning process across perception, prediction, and planning, and aligns it with the corresponding e2e module. Our approach VERDI harnesses language reasoning across perception, prediction, and planning, directly integrating it into the e2e pipeline, and does not require a VLM at runtime.

Knowledge Distillation, Representation Learning and Alignment. Knowledge distillation compresses and accelerates models by training a smaller "student" to mimic a larger "teacher," and has become essential as deep learning models continue to grow in size [41], [42]. In the era of large language models, it is particularly critical for transferring knowledge to more compact architectures and for boosting overall performance [43]. Our approach can be seen as a specialized form of distillation: rather than having a student network emulate the VLM's outputs directly, we train our modular end-to-end network to replicate the VLM's internal reasoning processes. This multi-modal aligned representation learning draws insight from contrastive learning [44], [45] and embedding space supervision [46]. Recently [47] has explored leveraging these pretrained embeddings by aligning them with learned modules for downstream control tasks, such as active exploration. Our method combines knowledge distillation with cross-modal alignment, embedding VLM reasoning into the driving model by aligning its textual and driving feature representations.

III. VLM-Embedded Reasoning

A. Problem Formulation

We cast the autonomous driving problem as future trajectory planning. Let $s_t = [x, y, z, v_x, v_y, v_z]_t$ be the ego agent state at time step t. The history trajectory of the past T time steps is denoted as $\tau_{-T:0} := \{s_{-T}, ..., s_0\}$, and the future trajectory with a planning horizon H is denoted $\tau_{0:H} := \{s_0, ..., s_H\}$. We have image observations at each time step t, denoted o_t. We aim to learn an end-to-end model $\mathcal{P}_\theta$, that plans future trajectories given only sequential observations:

$$\hat{\tau}_{0:H} \sim \mathcal{P}_\theta(\tau_{0:H}|o_{-T:0}), \tag{1}$$

where θ represents learnable weights of the model. As shown in Figure 1, our training pipeline focuses on the alignment of two modules: the VLM reasoning module, which we denote $\mathcal{M}$, and the e2e driving model, which we denote $\mathcal{P}$. Further, we assume that the modular e2e model can be broken down into three submodules $\mathcal{P}^i_\theta$, where $i \in \{\texttt{perception}, \texttt{prediction}, \texttt{planning}\}$. In addition to the standard losses for AD (e.g., ℓ_2 distance), we add a training objective to minimize the difference between all VLM reasoning steps and the intermediate e2e submodules. Our key idea is to accomplish this by aligning the text features from the VLM reasoning module, $f_\mathcal{M}$, with the driving features from the e2e model, $f_\mathcal{P}$, through a similarity loss in the overall loss function, which we describe in Section III-C. We detail how we obtain language embeddings in Section III-B, how we augment the e2e model by alignment with VLM embeddings in Section III-C.

B. Obtaining Language Embeddings

In this section, we describe how we acquire language features $f_\mathcal{M}$ from the VLM reasoning module $\mathcal{M}$. This module consists of two steps: (1) querying the VLM to obtain text responses, and (2) mapping the text response to latent features. Figure 2 outlines the VLM reasoning module.

Prompting Strategy. Following the modular e2e architecture, we prompt the VLM to reason about the perception, prediction, and planning steps in a chain-of-thought fashion [26]. We first guide the VLM with some system prompt to ensure its responses are concise and relevant. For perception, the VLM is asked to explain the past trajectory. It is given multi-view images captured from six cameras at timesteps $t = -T$ and $t = 0$, as well as the past ego trajectory $\tau_{-T:0}$. The VLM is then asked to identify the number of lanes in the scene, any agents or notable objects present, as well as their relative locations. To prevent spatial confusion and redundancy, the VLM is asked to label the agents and objects from left to right and from front to back. For prediction and planning, the VLM is asked to explain the future trajectory. In both queries, it is given the future front images at $t = 0$ and $t = H$, as well as the future trajectory $\tau_{0:H}$. For prediction, the answer from perception is provided and the VLM is asked to describe other agents' actions in the same order they were listed. The planning query in addition takes the prediction answers, and asks the VLM to describe what actions the ego agent took. The full prompt is listed in Appendix B-A.

Encoding VLM Responses. We then map the answer for each module to latent feature space, using a text encoder. The text encoder is able to map natural language to semantically meaningful sentence embeddings [48]. This encoding process reduces the dimensionality of the text, makes it machine interpretable and allows for alignment while preserving semantic meaning. The encoded text results in a feature vector $f^i_\mathcal{M}$ for each module i.

C. Obtaining Driving Model Alignment with VLM Reasoning

To obtain driving features $f^i_\mathcal{P}$ for each module i of the e2e model, we project its output features $F_i = \mathcal{P}^i_\theta(\cdot)$ into the shared latent space with $f^i_\mathcal{M}$ via a learnable Progressive Feature Projector (PFP) ϕ_i, producing the e2e feature $f^i_\mathcal{P} = \text{PFP}\phi_i(F_i)$ (more details in Appendix B-B). Figure 3 shows the training architecture that aligns the e2e model and VLM reasoning.

VLM-Distilled Scene Understanding. To enhance the precision of the Bird's Eye View (BEV) map's representation of the driving scene, we find it essential to extract high-quality visual features $F_{\texttt{perception}}$. Vision-Language Models (VLMs), trained on large-scale internet image-text pairs, possess strong capabilities in identifying object classes, road markings, and

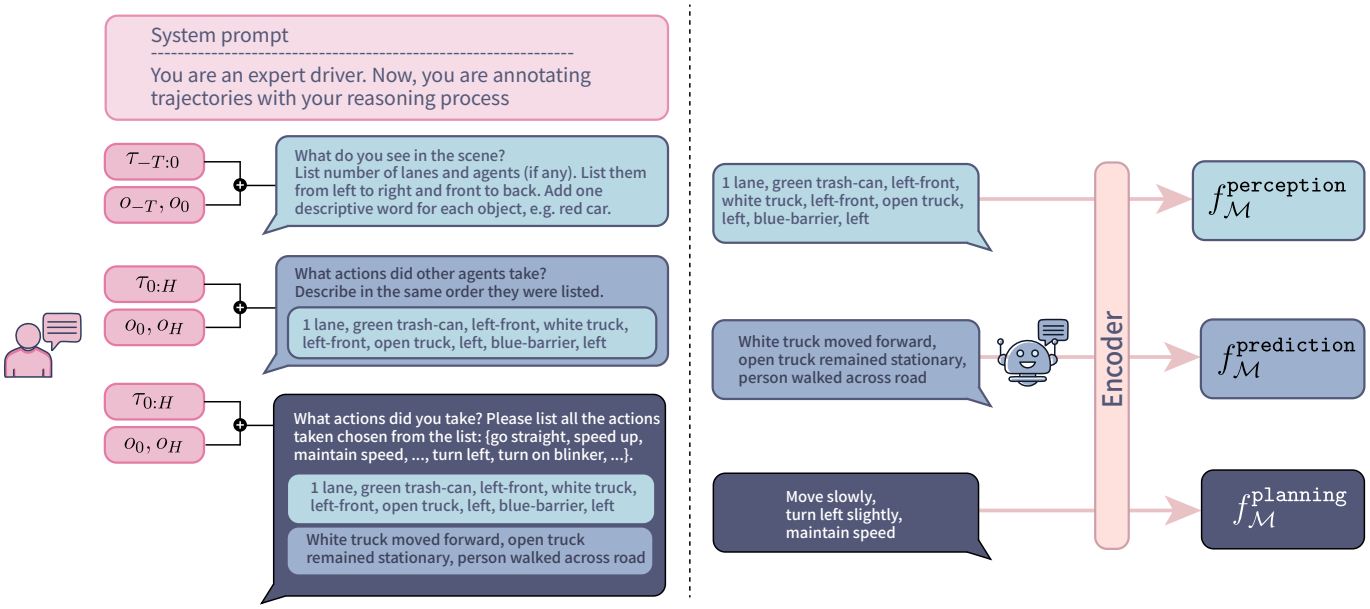


Fig. 2: Obtaining description features through chain-of-thought prompting and text encoder. For each query, the prompt consists of the system prompt, the observed images, the ego vehicle trajectory, the respective question, as well as the answers to the upstream modules (if any). The VLM answers to each module are encoded and mapped to a latent feature space.

spatial relationships with respect to the ego vehicle. Our goal is to distill this semantic understanding from the VLM into the image features $F_{\texttt{perception}}$ encoded by the image feature encoder, as presented in Figure 3. The features are then processed by the BEV Encoder to produce a refined BEV representation. To achieve alignment, we project $F_{\texttt{perception}}$ to $f_{\mathcal{P}}^{\texttt{perception}}$ in the shared latent space with $\text{PFP}_{\phi}^{\texttt{perception}}$, which consists of a combination of CNN and MLP layers(See Appendix B-C for details). The perception driving features are then ready to be aligned with VLM reasoning features.

VLM-Distilled Agents Prediction. Next, the end-to-end prediction module forecasts the next H timesteps from the BEV feature map $F_{\texttt{bev}}$. In Figure 3, $F_{\texttt{bev}}$ is decoded into agent and map queries, $Q_{\texttt{agents}}$ and $Q_{\texttt{map}}$, which presents a common architecture among e2e models [10], [11], [30]. These two separate feature maps are then concatenated to form the unified prediction query

$$Q_{\texttt{predict}} = \left[Q_{\texttt{agents}}; Q_{\texttt{map}}\right],$$

where $[\cdot;\cdot]$ denotes channel-wise concatenation. Here, $Q_{\texttt{agents}}$ encodes agent-agent relative positions and $Q_{\texttt{map}}$ encodes agent-map spatial relations.

To imbue the prediction module with the VLM's reasoning ability, we distill the VLM's reasoning ability by aligning VLM's prediction $f_{\mathcal{M}}^{\texttt{prediction}}$ and $f_{\mathcal{P}}^{\texttt{prediction}}$, which we obtain by projecting $Q_{\texttt{predict}}$ through an MLP-based $\text{PFP}_{\phi}^{\texttt{prediction}}$.

VLM Distilled Ego Planning. We feed the combined query $Q_{\texttt{predict}}$, which integrates both other agents' future trajectories and spatial map context, into the planning module $\mathcal{P}_{\texttt{planning}}$. Conditioned on $Q_{\texttt{predict}}$, the planner produces the latent ego feature $Q_{\texttt{ego}}$ as:

$$Q_{\texttt{ego}} = \mathcal{P}_{\theta}^{\texttt{planning}}(\tau_{0:H}|Q_{\texttt{predict}}). \tag{2}$$

We project $Q_{\texttt{ego}}$ onto the latent space with an MLP based $\text{PFP}_{\phi}^{\texttt{planning}}$, before passing it into the last-layer decoder. This allows us to enrich $Q_{\texttt{ego}}$ with VLM reasoning by aligning VLM's $f_{\mathcal{M}}^{\texttt{planning}}$ with the latent e2e planning features $f_{\mathcal{P}}^{\texttt{planning}}$.

Alignment Loss We use cosine similarity to represent the feature alignment loss $\mathcal{L}_f(f_{\mathcal{P}}, f_{\mathcal{M}})$ between the e2e features and the language features,

$$\mathcal{L}_f(f_{\mathcal{P}}, f_{\mathcal{M}}) = \frac{f_{\mathcal{P}} \cdot f_{\mathcal{M}}}{\|f_{\mathcal{P}}\|\|f_{\mathcal{M}}\|}, \text{ with a total loss} \tag{3}$$

$$\mathcal{L}_i = \mathcal{L}_e(\theta_i) + \lambda_i \mathcal{L}_f(f_{\mathcal{P}}^i, f_{\mathcal{M}}^i), \tag{4}$$

where $\mathcal{L}_e(\theta_i)$ represents the supervised loss function with the original ground-truth data according to [10] and λ is a weight hyperparameter balancing the two contributions.

IV. Experiments

We confirm the efficacy of VERDI through a series of experiments, comparing against baseline approaches in Section IV-A, and validate all design choices with ablation experiments in Section IV-C.

Baseline Approaches. We compare our method to two main categories of baselines: (1) e2e methods that directly train a modular differentiable network for future trajectory planning, (2) methods that train or finetune a VLM to infer future actions

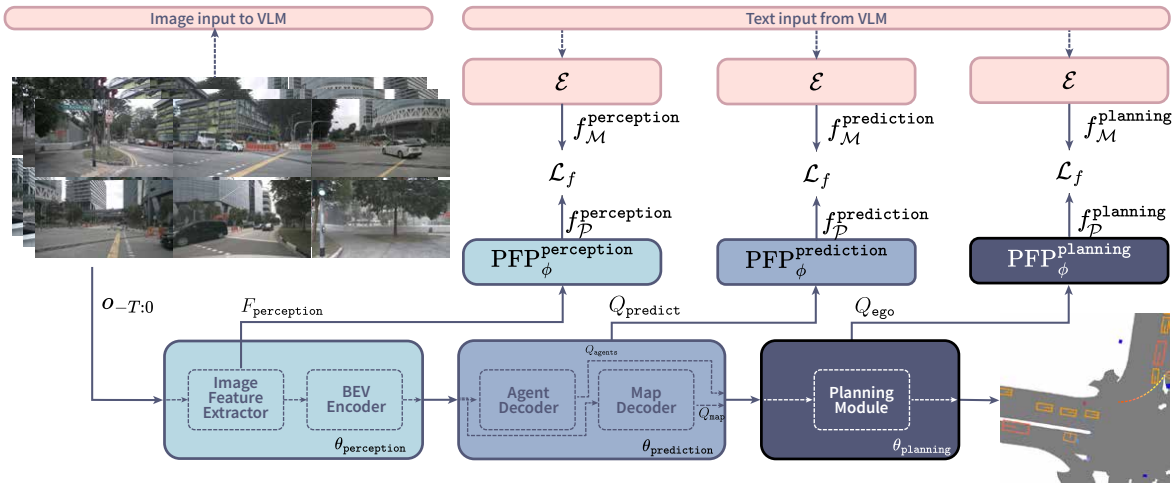


Fig. 3: **VERDI Training.** The e2e model is trained with VERDI for the individual perception, prediction, and planning modules. All relevant feature maps F and Q are first mapped to a feature $f_{\mathcal{P}}$ in a representation space, which is shared with the encoded language features $f_{\mathcal{M}}$. This mapping is facilitated by VERDI's trainable PFP layers. The perception outputs $F_{\texttt{perception}}$ including the extracted image features, are directly supervised with the encoded VLM features. In the subsequent modules, all features are supervised. $\mathcal{L}_f$ computes their similarity.

TABLE I: **Evaluation on the nuScenes dataset** [13]. Methods are compared according to: (1) Whether a VLM is required at inference; (2) Inference speed (FPS); (3) Trajectory accuracy, measured as the ℓ_2 distance to the expert trajectory at 1s, 2s, and 3s horizon; and (4) Whether precise historical ego-vehicle state is used in planning. In a fair co10mparison with methods not having privileged access to ego status, including location, VERDI achieves the best performance across all metrics.

Method	Requires VLM @ Inference	FPS ↑	l_2 (1s) ↓	l_2 (2s) ↓	l_2 (3s) ↓	l_2 (avg.) ↓	Ego Status
DriveVLM [25]	✓	2.43	0.18	0.34	0.68	0.40	✓
OpenEMMA [19]	✓	NA	1.45	3.21	3.76	2.81	✓
OmniDrive [22]	✓	0.44	1.15	1.96	2.84	1.98	-
UniAD [30]	-	1.8	0.48	0.96	1.05	0.83	-
VAD-Base [10]	-	**4.5**	0.41	0.70	1.05	0.72	-
VERDI	-	**4.5**	**0.36**	**0.62**	**0.96**	**0.65**	-

or control signals. We implement our method building on the e2e model VAD-Base [10], making it a direct baseline. We also compare against UniAD [9] , a representative state-of-the-art e2e method. For VLM-based methods, we compare against DriveVLM [25], which generates chain-of-thought reasoning and hierarchical plans using a VLM, OmniDrive [22], which aligns 3D BEV features with counterfactual scene reasoning via a fine-tuned VLM, and OpenEMMA [19], an open-source implementation of Waymo's EMMA, predicting full planner trajectories using a fine-tuned VLM.

Dataset and Metrics. We conduct experiments on the nuScenes dataset [13], which contains approximately 1000 driving scenes, each about 20 seconds long. The dataset includes 6 camera views, and provides annotated keyframes at 2Hz. For planning evaluation, we follow the standard open-loop protocol [30]: given 2 seconds of past observations, the model predicts the vehicle's trajectory over the next 3 seconds. Performance is measured using ℓ_2 displacement errors at 1, 2, and 3-second horizons. We also compare inference speed in frames per second (FPS) for applicable methods.

Implementation Details. Training was conducted for 60 epochs on 10 NVIDIA A6000 GPUs with a batch size of 2 using the AdamW optimizer. Inference was done on a single NVIDIA A6000 GPU for all models. All other training configurations maintain the settings adopted from VAD [10]. For VLM language embeddings, we use `Qwen-2.5-VL-72B` [49] to annotate every trajectory in the training dataset. We use the chain-of-thought prompting strategy described in Section III-B to generate text outputs from multi-view images and states, and map the text onto embedding space using the `all-mpnet-base-v2` sentence-transformers model [48], [50].

A. Quantitative Assessment

Table I summarizes our quantitative experimental findings. Note that we separately evaluate methods that condition trajectory planning on *history ego status – privileged information that typically perform better [12] – and real-time deployable methods that do not have access to the precise ego status.* Among baselines that do not use ego status, VERDI achieves the lowest average ℓ_2 distance to the expert trajectory (0.65)—an 10% improvement over our direct baseline, VAD-

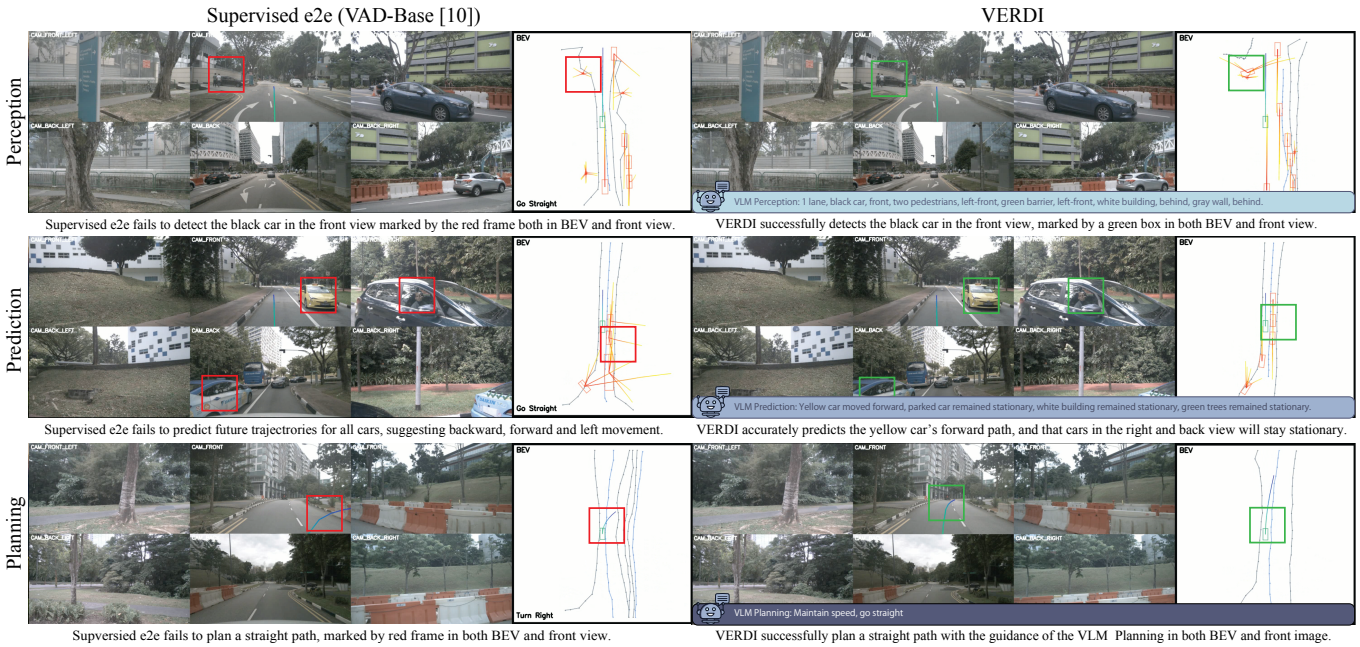


Fig. 4: Qualitative comparison of VERDI (Ours, right column) and the Supervised e2e model (baseline, left column) on the nuScenes dataset [13]. Each entry shows the multi-view camera observations on the left and the BEV view on the right at one time step t. The left panel overlays the ego agent's planned 3-second trajectory on the front-camera image and BEV panel as a solid green line that fades to blue. The BEV panel renders the ego vehicle as a green rectangle, pedestrians and other vehicles as red rectangles, and their predicted 3-second trajectories as red lines. Each of the example shows our successful performance on the perception, prediction, and planning modules, indicated by □, while failures are highlighted by □. We also show the VLM text response for each testing case to demonstrate that VERDI has successfully distilled VLM's reasoning and commonsense knowledge.

Base. While DriveVLM—leveraging ego status—attains a lower ℓ_2 distance, VERDI runs ~ 1.9 times faster. Compared to baseline methods that report inference speed (e.g., 2.44 FPS for DriveVLM [25], 0.44 FPS for OmniDrive implemented with Pytorch [22]). Overall, VERDI offers the best trade-off between inference speed and trajectory accuracy among methods without ego status.

B. Qualitative Assessment

We present qualitative comparisons between the Supervised e2e (VAD-Base) (left column) and VERDI(right column) in Figure 4. Our approach yields better perception results: for example, in the top row, the supervised e2e detects only the pedestrians with red sparkle sign and fails to localize the front-left vehicle in the BEV view □, whereas our model correctly identifies both the pedestrians and that vehicle □. Detecting this vehicle is critical as omitting it could lead to a cross-traffic collision. VERDI also has more accurate future motion prediction. In the second row, the ego vehicle continues straight, while the front-right taxi, the adjacent black car, and the rear white-and-blue taxi all wait at the traffic light, indicating they will likewise proceed straight once it turns green. VERDI correctly predicts these straight-ahead trajectories shown as red lines aligned with each vehicle's heading, whereas the supervised e2e baseline mistakenly forecasts leftward and backward movements. This confirms that the VLM commonsense reasoning substantially improves the accuracy of predictions on other agents' motion. In the third row, VERDI produces a mostly straight trajectory with a slight rightward adjustment, as shown in both the front-view and BEV projections. In contrast, the Supervised e2e baseline chooses a sharp right turn that would collide with the barrier.

C. Ablation Study

We conduct ablation experiments to analyze the contribution of each component. All model derivations are trained based on VAD-tiny with a sub-sampled set of nuScenes, which we describe in more detail in Appendix C.

Alignment Module Variations. In Table II, we show that each alignment module improves the performance at varying rates. To evaluate models trained with aligning only certain modules p, we simply set $\lambda_i = 10$ for $i = p$ and $\lambda_j = 0$ for $j \neq p$. Notably, aligning only features of the perception module improves ℓ_2 loss in the short term (1 second), while aligning only the prediction and planning modules improves ℓ_2 loss in the long term (3 seconds). Aligning with all three achieves a desired balance of short and long-term objectives, resulting in a low average ℓ_2 loss. This result is consistent with the way we split the trajectories into past and future to align with each module, causing alignment with only perception to be short-sighted. Aligning with the planning module alone achieves

TABLE II: **Ablations Module Alignment**. Results on aligning different e2e modules. Bolded numbers stand for the best and italic numbers stand for second best.

Aligned Modules			ℓ_2 error ↓			
Perception	**Prediction**	**Planning**	(1s)	(2s)	(3s)	(avg.)
			1.23	2.07	2.94	2.08
✓			**1.14**	1.91	2.70	1.92
	✓		1.17	1.95	2.77	1.96
		✓	1.15	**1.90**	2.70	1.92
✓	✓		1.19	1.99	2.82	2.00
✓		✓	1.20	1.97	2.76	1.98
	✓	✓	1.16	**1.90**	**2.66**	**1.90**
✓	✓	✓	**1.14**	**1.90**	*2.69*	*1.91*

TABLE III: **Ablations VLM embeddings**. Ablation results with VLM embeddings of varying quality.

VLM Embeddings	ℓ_2 (1s) ↓	ℓ_2 (2s) ↓	ℓ_2 (3s) ↓	ℓ_2 (avg.) ↓
Baseline (no VLM)	1.23	2.07	2.94	2.08
Adversarial VLM	1.22	2.04	2.88	2.05
QwenVL-7B-CLIP	1.20	2.01	2.84	2.02
QwenVL-72B-CLIP	1.19	1.99	2.84	2.01
QwenVL-72B-ST-mini	1.21	2.04	2.89	2.05
QwenVL-72B-ST-base	**1.15**	**1.93**	**2.74**	**1.94**

good performance, possibly because planning reasoning uses a query containing perception and prediction data. We provide more training details in Appendix C-A, showing the loss curve saturating.

VLM Embedding Variations. We investigate the role of VLM embeddings and ablate our model by providing VLM embeddings of varying qualities. In Table III, we report that using a higher-quality VLM (Qwen-72B rather than Qwen-7B) improves performance. The choice of the text encoder and hence the shared latent space impacts performance. Further, we find that using CLIP [44] as text encoder yields worse performance than the `all-mpnet-base-v2` sentence-transformers model, since bag of words used by CLIP leads to less semantically meaningful embeddings. Moreover, we evaluate with an adversarial VLM that generates false text embeddings and aligns the adversarial text features with the driving features. To generate the false texts, we shuffle the true perception and prediction VLM embeddings, as well as the counterfactual responses for planning (e.g., "speed up, turn right abruptly, run into the barrier"). With this set of VLM embeddings, the performance of the e2e model remains mostly unchanged from the baseline method. We hypothesize that this is because the e2e model learns to ignore VLM information in this scenario.

V. Conclusion

We present VERDI, a training-time method that distills a Vision–Language Model reasoning into a lightweight end-to-end autonomous-driving stack by aligning each submodule with VLM reasoning embeddings. Through extensive experiments and ablation studies on nuScenes, VERDI achieves a 10% performance gain in ℓ_2 distance over the supervised baseline without reasoning embeddings. Although our unoptimized implementation runs at 4.5 Hz, it has not yet been accelerated with TensorRT or additional compression, leaving substantial headroom for faster inference in practical deployments. In the future, we plan to not only extend the method to additional vision tasks, such as depth and occupancy prediction, but also provide supervision with closed-loop simulation, potentially mutually improving the aligned driving model and VLM in tandem.

VI. Limitations

While VERDI presents a promising approach to distilling VLM reasoning into modular autonomous driving systems, several limitations remain.

Reasoning-Driving Alignment Loss. VERDI shows that aligning the VLM's textual reasoning with intermediate module outputs effectively transfers semantic understanding. However, this alignment relies on the design of the cosine similarity loss across features, which may fail to capture finer-grained semantics. Future work could explore more expressive alignment objectives or incorporate contrastive learning strategies that better preserve more nuanced structural reasoning.

Pretrained VLMs. Our results depend on the capabilities of the VLM. Although we use `Qwen-2.5-VL-72B` due to its ease of access and satisfactory performance, we do notice that it sometimes inaccurately explains the reasoning processes. As shown in our ablations, lower-capacity VLMs lead to degraded performance, while adversarial or poorly aligned prompts can nullify the benefit of reasoning supervision. A promising direction would be to investigate more performant VLMs or task-specific finetuning to improve reasoning quality.

Multi-Modality Coverage. Currently, VERDI focuses on aligning visual and textual modalities but does not consider other sensory inputs such as LiDAR or radar. Additionally, the system only leverages VLM supervision during training, and does not adapt post-deployment. Future work could explore continual learning or co-training frameworks where the driving model and VLM can mutually refine each other through interaction.

Evaluation in Closed-Loop Simulation. VERDI is currently evaluated only in open-loop settings on the nuScenes dataset. This does not fully capture downstream planning performance, especially under compounding errors or interactions with dynamic agents. A natural next step is to evaluate VERDI in closed-loop simulators such as CARLA [51], which would enable systematic stress testing in diverse, interactive, and controllable environments. Such evaluation would also allow benchmarking VERDI's ability to generalize across different cities, weather conditions, and corner-case driving scenarios.

Acknowledgments

Zhiting Mei and Anirudha Majumdar were partially supported by the NSF CAREER Award #2044149, the Office of Naval Research (N00014-23-1-2148), and a Sloan Fellowship. Felix Heide was supported by an NSFCAREER Award #2047359, a Packard Foundation Fellowship, a Sloan

Research Fellowship, a Sony Young Faculty Award, a Project XInnovation Award, and an Amazon Science Research Award. The authors would like to thank Mario Bijelic for helpful discussions, paper editing and proofread on this work.

REFERENCES

[1] L. Di Lillo, T. Gode, X. Zhou, M. Atzei, R. Chen, and T. Victor, "Comparative safety performance of autonomous-and human drivers: A real-world case study of the waymo driver," *Heliyon*, vol. 10, no. 14, 2024.

[2] M. Jung, J. Park, and M.-S. Pang, "Safety on autopilot: An empirical investigation of autonomous driving and traffic safety," 2024.

[3] D. Chen and P. Krähenbühl, "Learning from all vehicles," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 17 222–17 231.

[4] X. Liang, Y. Wu, J. Han, H. Xu, C. Xu, and X. Liang, "Effective adaptation in multi-task co-training for unified autonomous driving," *Advances in Neural Information Processing Systems*, vol. 35, pp. 19 645–19 658, 2022.

[5] W. Zeng, W. Luo, S. Suo, A. Sadat, B. Yang, S. Casas, and R. Urtasun, "End-to-end interpretable neural motion planner," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8660–8669.

[6] Y. Zhang, Z. Zhu, W. Zheng, J. Huang, G. Huang, J. Zhou, and J. Lu, "Beverse: Unified perception and prediction in birds-eye-view for vision-centric autonomous driving," *arXiv preprint arXiv:2205.09743*, 2022.

[7] L. Zhang, Y. Xiong, Z. Yang, S. Casas, R. Hu, and R. Urtasun, "Learning unsupervised world models for autonomous driving via discrete diffusion," *arXiv preprint arXiv:2311.01017*, 2023.

[8] X. Zhang, X. Tan, Y. An, Y. Li, and Z. Fan, "Oatracker: Object-aware anti-occlusion 3d multiobject tracking for autonomous driving," *Expert Systems with Applications*, vol. 252, p. 124158, 2024.

[9] Y. Hu, J. Yang, L. Chen, K. Li, C. Sima, X. Zhu, S. Chai, S. Du, T. Lin, W. Wang *et al.*, "Planning-oriented autonomous driving," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 17 853–17 862.

[10] 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," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 8340–8350.

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

[12] Z. Li, Z. Yu, S. Lan, J. Li, J. Kautz, T. Lu, and J. M. Alvarez, "Is ego status all you need for open-loop end-to-end autonomous driving?" in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 14 864–14 873.

[13] 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," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 621–11 631.

[14] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine *et al.*, "Scalability in perception for autonomous driving: Waymo open dataset," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2446–2454.

[15] J.-J. Hwang, R. Xu, H. Lin, W.-C. Hung, J. Ji, K. Choi, D. Huang, T. He, P. Covington, B. Sapp *et al.*, "Emma: End-to-end multimodal model for autonomous driving," *arXiv preprint arXiv:2410.23262*, 2024.

[16] H. A. Simon, "Bounded rationality," *Utility and probability*, pp. 15–18, 1990.

[17] B. D. Jones, "Bounded rationality," *Annual review of political science*, vol. 2, no. 1, pp. 297–321, 1999.

[18] J.-J. Hwang, R. Xu, H. Lin, W.-C. Hung, J. Ji, K. Choi, D. Huang, T. He, P. Covington, B. Sapp, J. Guo, D. Anguelov, and M. Tan, "EMMA: End-to-End Multimodal Model for Autonomous Driving," Oct. 2024, arXiv:2410.23262 [cs] version: 1. [Online]. Available: http://arxiv.org/abs/2410.23262

[19] S. Xing, C. Qian, Y. Wang, H. Hua, K. Tian, Y. Zhou, and Z. Tu, "OpenEMMA: Open-Source Multimodal Model for End-to-End Autonomous Driving," Dec. 2024, arXiv:2412.15208 [cs]. [Online]. Available: http://arxiv.org/abs/2412.15208

[20] L. Chen, O. Sinavski, J. Hünermann, A. Karnsund, A. J. Willmott, D. Birch, D. Maund, and J. Shotton, "Driving with llms: Fusing object-level vector modality for explainable autonomous driving," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 14 093–14 100.

[21] Z. Xu, Y. Zhang, E. Xie, Z. Zhao, Y. Guo, K.-Y. K. Wong, Z. Li, and H. Zhao, "Drivegpt4: Interpretable end-to-end autonomous driving via large language model," *IEEE Robotics and Automation Letters*, 2024.

[22] S. Wang, Z. Yu, X. Jiang, S. Lan, M. Shi, N. Chang, J. Kautz, Y. Li, and J. M. Alvarez, "Omnidrive: A holistic llm-agent framework for autonomous driving with 3d perception, reasoning and planning," *arXiv preprint arXiv:2405.01533*, 2024.

[23] C. Sima, K. Renz, K. Chitta, L. Chen, H. Zhang, C. Xie, J. Beißwenger, P. Luo, A. Geiger, and H. Li, "Drivelm: Driving with graph visual question answering," in *European Conference on Computer Vision*. Springer, 2024, pp. 256–274.

[24] W. Wang, J. Xie, C. Hu, H. Zou, J. Fan, W. Tong, Y. Wen, S. Wu, H. Deng, Z. Li *et al.*, "Drivemlm: Aligning multi-modal large language models with behavioral planning states for autonomous driving," *arXiv preprint arXiv:2312.09245*, 2023.

[25] X. Tian, J. Gu, B. Li, Y. Liu, Y. Wang, Z. Zhao, K. Zhan, P. Jia, X. Lang, and H. Zhao, "Drivevlm: The convergence of autonomous driving and large vision-language models," *arXiv preprint arXiv:2402.12289*, 2024.

[26] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in neural information processing systems*, vol. 35, pp. 24 824–24 837, 2022.

[27] L. Chen, P. Wu, K. Chitta, B. Jaeger, A. Geiger, and H. Li, "End-to-end autonomous driving: Challenges and frontiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[28] P. S. Chib and P. Singh, "Recent advancements in end-to-end autonomous driving using deep learning: A survey," *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 1, pp. 103–118, 2023.

[29] P. Karkus, B. Ivanovic, S. Mannor, and M. Pavone, "Diffstack: A differentiable and modular control stack for autonomous vehicles," in *Conference on robot learning*. PMLR, 2023, pp. 2170–2180.

[30] 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," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[31] H. Shao, L. Wang, R. Chen, H. Li, and Y. Liu, "Safety-enhanced autonomous driving using interpretable sensor fusion transformer," in *Conference on Robot Learning*. PMLR, 2023, pp. 726–737.

[32] Z. Yang, L. Chen, Y. Sun, and H. Li, "Visual point cloud forecasting enables scalable autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 14 673–14 684.

[33] Y. Yang, B. Feng, K. Wang, N. Leonard, A. B. Dieng, and C. Allen-Blanchette, "Behavior-inspired neural networks for relational inference," *arXiv preprint arXiv:2406.14746*, 2024.

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

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

[36] S. Biswas, S. Casas, Q. Sykora, B. Agro, A. Sadat, and R. Urtasun, "Quad: Query-based interpretable neural motion planning for autonomous driving," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 14 236–14 243.

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

[38] S. Doll, N. Hanselmann, L. Schneider, R. Schulz, M. Cordts, M. Enzweiler, and H. P. Lensch, "Dualad: Disentangling the dynamic and static world for end-to-end driving," in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 14 728–14 737.

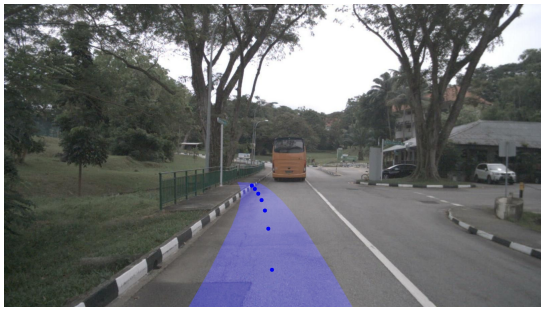
[39] X. Weng, B. Ivanovic, Y. Wang, Y. Wang, and M. Pavone, "Paradrive: Parallelized architecture for real-time autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 15 449–15 458.

[40] C. Pan, B. Yaman, T. Nesti, A. Mallik, A. G. Allievi, S. Velipasalar, and L. Ren, "Vlp: Vision language planning for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 14 760–14 769.
[41] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.
[42] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge distillation: A survey," *International Journal of Computer Vision*, vol. 129, no. 6, pp. 1789–1819, 2021.
[43] X. Xu, M. Li, C. Tao, T. Shen, R. Cheng, J. Li, C. Xu, D. Tao, and T. Zhou, "A survey on knowledge distillation of large language models," *arXiv preprint arXiv:2402.13116*, 2024.
[44] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning Transferable Visual Models From Natural Language Supervision," in *Proceedings of the 38th International Conference on Machine Learning*. PMLR, Jul. 2021, pp. 8748–8763, iSSN: 2640-3498. [Online]. Available: https://proceedings.mlr.press/v139/radford21a.html
[45] M. Cherti, R. Beaumont, R. Wightman, M. Wortsman, G. Ilharco, C. Gordon, C. Schuhmann, L. Schmidt, and J. Jitsev, "Reproducible Scaling Laws for Contrastive Language-Image Learning," 2023, pp. 2818–2829. [Online]. Available: https://openaccess.thecvf.com/content/CVPR2023/html/Cherti_Reproducible_Scaling_Laws_for_Contrastive_Language-Image_Learning_CVPR_2023_paper
[46] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, "Supervised contrastive learning," *Advances in neural information processing systems*, vol. 33, pp. 18 661–18 673, 2020.
[47] C. Sancaktar, C. Gumbsch, A. Zadaianchuk, P. Kolev, and G. Martius, "Sensei: Semantic exploration guided by foundation models to learn versatile world models," *arXiv preprint arXiv:2503.01584*, 2025.
[48] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," *arXiv preprint arXiv:1908.10084*, 2019.
[49] A. Yang, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Li, D. Liu, F. Huang, H. Wei *et al.*, "Qwen2. 5 technical report," *arXiv preprint arXiv:2412.15115*, 2024.
[50] "sentence-transformers/all-mpnet-base-v2 · Hugging Face," Jan. 2024. [Online]. Available: https://huggingface.co/sentence-transformers/all-mpnet-base-v2
[51] 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.
[52] Z. Li, W. Wang, H. Li, E. Xie, C. Sima, T. Lu, Q. Yu, and J. Dai, "Bevformer: learning bird's-eye-view representation from lidar-camera via spatiotemporal transformers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

APPENDIX A
INFERENCE TIME VLM HALLUCINATION

Methods that use finetuned multimodal VLMs at inference time are subject to the risk of hallucinations, which largely stem from the VLMs' limited spatial reasoning capabilities, as discussed in Section 1.0. Since these models are trained almost exclusively on 2D image data, they lack understanding of depth cues provided by modalities like LiDAR. As a result, VLMs often misjudge object orientation and distance, producing descriptions that contradict the scene's geometry. We show three examples of VLM hullucination from OpenEMMA [19], an open-source implementation of EMMA [15]. OpenEMMA uses the front image as the only input modality to the VLM and uses the chain of the thought to reason about the scenes, driver intent, and planning decisions.

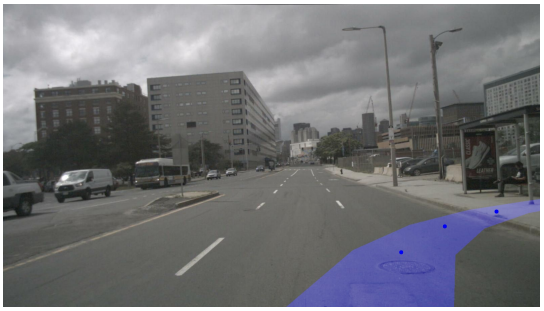
In Figure 5, an orange bus is driving directly in front of the ego vehicle, traveling in the same direction. However, the VLM's response for the "Bus Scene" (highlighted in red) involves three clear hallucinations. First and most importantly, it wrongly describes the bus as moving toward the viewer on the same road. Second, it invents a dashed line on the left side—there is actually only a painted curb. Third, it self-contradicts by stating that no other cars or pedestrians are visible, despite the parked cars in the upper right corner and the bus itself clearly appearing in the image. The planned intent—a slight left turn—would clearly result in a collision with the curb and shoulder. Because the VLM relies only on a single 2D image without any depth cues or a reconstructured BEV view, it lacks the spatial reasoning needed to anticipate such obstacles, undermining both the accuracy and interpretability



OpenEMMA Response (Bus Scene)

```
Scene Description: As an autonomous driving
labeller, I would describe the driving
scene based on the provided image and the
information about the camera's history.
Here is the description:
1. **Traffic Lights**: There are no visible
traffic lights in the image. 2. **Movements
of Other Cars or Pedestrians**: There are
no other cars or pedestrians visible in
the image. 3. **Lane Markings**: The road
has clear lane markings, including a solid
white line on the right side and a dashed
white line on the left side. There is also
a black and white striped curb on the left
side of the road.
The image shows a quiet Object Description:
In the driving scene, there are several
road users that you should pay attention
to:
1. **Bus (Location: Center of the
image, moving towards the camera)**: -
**Description**: The bus is moving towards
you on the same road. It is important to
monitor its speed and direction to ensure
safe overtaking or passing. - **Why it's
important**: Ensuring you have enough space
to overtake safely is crucial to avoid
collisions. ...
```

Fig. 5: OpenEMMA Testing Example (Bus Scene) on the nuScenes Dataset [13]. In the front-view image, OpenEMMA's projected future path is overlaid as light blue. An orange bus occupies the same lane, traveling in the same direction as the ego vehicle. Solid white lane markings run along the right side, with a white-striped curb on the left. OpenEMMA erroneously plans a leftward trajectory, which would result in a collision with that curb.



OpenEMMA Response(City Scene)

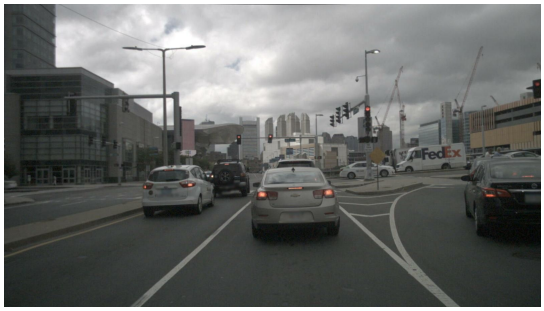
```
Scene Description: As an autonomous driving
labeller, I would describe the driving
scene based on the provided images and
the information available from the camera
at a 0.5-second interval over the past 5
seconds.
Current Frame (0.5 seconds ago): -
**Traffic Lights:** Not visible in the
current frame. - **Other Cars:** - A white
van is driving on the left side of the road,
moving in the same direction as the viewer.
- A bus is also driving on the left side of
the road, moving in the same direction as
the viewer. - Several cars are visible on
the right side of the ...
```

Fig. 6: OpenEMMA Testing Example (City Scene) on the nuScenes Dataset [13]. In the front-view image, OpenEMMA's predicted trajectory is overlaid in light blue. The ego vehicle's current lane is clear of obstacles or other traffic. On the left, a white van, a bus, and several smaller cars approach in the opposite direction in the adjacent lane. Despite this clear path, OpenEMMA incorrectly plans a rightward turn—risking a potential collision.

of using its output directly for planning.

In Figure 6, the ego vehicle travels alone down the road with no obstacles directly ahead. On the opposite side of the road, a white bus, a white van, and several smaller cars approach from the left in the oncoming lane. However, the VLM's response for the "City Scene" wrongly reports that the bus and van are both on the left side of the road and moving in the same direction as the ego vehicle, an error that causes the planned trajectory to steer inaccurately to the right.

In Figure 7, the ego vehicle and surrounding cars have all stopped at the red light and are waiting for it to turn green, so no trajectory is planned. However, OpenEMMA's response surprisingly highlights the white FedEx van on the right, presumably because it appears near the image center and thus seems close enough to require extra braking distance. A human observer, on the other hand, would immediately recognize that this van poses no real threat at its actual distance. This misattention stems from the model's lack of explicit depth perception and spatial reasoning, which leads it to hallucinate a hazard where none exists.



OpenEMMA Response (Traffic light Scene)

```
Scene Description: As an autonomous driving
labeller, I would describe the driving
scene based on the provided images and
the information available from the camera
at a 0.5-second interval over the past 5
seconds.
Current Frame (0.5 seconds ago): -
**Traffic Lights**: The traffic light in
the center of the image is red, indicating
that vehicles must stop. - **Other Cars**: -
The car directly in front of you is a silver
sedan, which appears to be stationary. - To
the left, there is a white sedan that is
also stationary. - To the right, there is a
black sedan that is Object Description: In
the driving scene, there are several road
users that you should pay attention to:
1. **FedEx Truck (Location: Right side
of the image, near the center)** -
**Description:** The FedEx truck is on the
right side of the image, near the center.
It is important to pay attention to this
truck because it is a large vehicle that may
have a longer stopping distance than smaller
cars. You should be prepared for it to slow
down or stop suddenly, especially if it is
making a turn or changing lanes. ...
```

Fig. 7: OpenEMMA Testing Example (Stop Sign Scene) on the nuScenes Dataset [13]. In the front-view image, all cars in front stop for the red traffic lights. There is no trajectory planne dand projected onto the front image. Notice, the white FeDex car on the right corner. Due to the lack of the depth and spatial understandingting, openEMMA suggested to keep an eye on this FeDex van since it thinks it could be a treat to the ego vehicle.

VERDI avoids single-view hallucinations by leveraging multi-view imagery and constructing a BEV map. From the multi-view image set, we generate a BEV representation that encodes spatial constraints, including lanes, road boundaries, other agents, and obstacles, ensuring accurate geometric context for downstream planning. Simultaneously, VERDI distills the VLM's strong visual recognition to identify scene objects and align their textual embeddings with the projected multi-view image features, producing dense, semantically informed features that feed into the BEV encoder.

APPENDIX B
IMPLEMENTATION DETAILS

A. Full Prompt for VLM Queries (Section 3.2 in main).

Below, we show the full chain-of-thought prompts used to acquire VLM reasoning embeddings. The system prompt is parsed for every prompt, and the perception, prediction, and planning prompts shown below take inputs from answers to previous modules.

System Prompt

```
You are an expert driver collecting data
in various scenarios for a self-driving
car. Now, you are annotating the collected
trajectories by providing concise
and accurate descriptions as short,
comma-separated sentences.
```

Perception

```
The images shown are the multi-views of a
driving scenario at the beginning and end
of the trajectory, in the order of [front
left, front, front right, back left, back,
back right]. The past {T} time steps (0.5
seconds each) of the vehicle states, in the
form of [x, y, z, vx, vy, vz], is given as:
{...}

What do you see in the scene? List the
following details in order: number of
lanes; agents (if any) chosen from
[car, truck, construction_vehicle, bus,
trailer, barrier, motorcycle, bicycle,
pedestrian, traffic_cone]; Location of the
objects (if any) chosen from [front, left,
left-front, left-behind, right, right-front,
right-behind, behind]; List them from left
to right and front to back. Only list the
objects that are visible in the images. Add
one descriptive word for each object, e.g.,
red car.
```

Prediction

```
The images shown are the front views of a
driving scenario at the beginning and end
of the trajectory. The past {T} time steps
(0.5 seconds each) of the vehicle states, in
the form of [x, y, z, vx, vy, vz], is given
as: {...}

What actions did other agents take? Describe
in the same order they were listed.

Perception: {answer to perception prompt}
```

Planning

```
The images shown are the front views of a
driving scenario at the beginning and end
of the trajectory. The past {T} time steps
(0.5 seconds each) of the vehicle states, in
the form of [x, y, z, vx, vy, vz], is given
as: {...}

What actions did you take? Please list
all the actions taken chosen from the
list: {go straight, speed up, maintain
speed, move slowly, stop smoothly, stop
abruptly, reverse, turn left slightly, turn
on blinker, turn right slightly, turn left
abruptly, turn right abruptly, turn around,
merge into the left lane, merge into the
right lane}

Perception: {answer to perception prompt}
Prediction: {answer to prediction prompt}
```

B. Progressive Feature Projector (PFP) for Perception

We introduce Progressive Feature Projectors (PFP) to progressively compress and project perception, prediction and planning features of end-to-end models After obtaining the results from implicit e2e features to language aligned feature spaces.

$\text{PFP}_{\phi}^{\texttt{perception}}$

	Layer	Activation	Output Dimension
0	Input: $\mathbf{F}^{\texttt{perception}}$	-	$\mathbb{R}^{B\times 6\times 256\times 12\times 20}$
1	Conv3D	ReLU	$\mathbb{R}^{B\times 256\times 1\times 12\times 20}$
2	$\text{Conv2D}_{s=2,p=1}$	ReLU	$\mathbb{R}^{B\times 256\times 6\times 10}$
3	$\text{Conv2D}_{s=2,p=1}$	ReLU	$\mathbb{R}^{B\times 256\times 3\times 5}$
4	$\text{Conv2D}_{s=1,p=1}$	ReLU	$\mathbb{R}^{B\times 256\times 3\times 5}$
5	Reshape	-	$\mathbb{R}^{B\times (256\cdot 3\cdot 5)}$
6	Linear	-	$\mathbb{R}^{B\times D}$

TABLE IV: **Architecture of the PFP for Perception**. To achieve alignment between language and perception features the presented architecture compresses and extracts language space-aligned information from the perception features of the e2e model.

Especially in the perception stage of autonmous driving e2e models feature maps, such as intermediate features of the BEV encoder, e.g. BEVformer [52] in VAD [10], become very large. For this stage we introduce a unique PFP to progressively compress the large feature map while preserving the feature information needed. We present an overview of the implementation used in VERDI in Tab. IV

Initially a batch of multi-view image features with the shape of $\mathbf{F_i}^{\texttt{perception}} \in \mathbb{R}^{B\times V\times C\times H\times W}$, where B is the batch size, $V = 6$ is the number of surrounding camera views, $C = 256$ is the channel dimension, and $H = 12$, $W = 20$ are the spatial dimensions after the image feature extraction. The Input features are then compressed across there view dimension using a 3D Convolution. Then the fused map is processed by a sequence of 2D convolutional layers to reduce

the spatial dimension and extract essential information. A final 2D convolution refines these coarse spatial features without further down-sampling. Finally, we reshape the feature map into a flat vector for the subsequential processing and apply a linear projection to obtain the final compressed feature. This feature will be used to align with the encoded feature map from VLM's annotation, where D represents the size of the shared feature space.

C. Fully Connected Layers (MLP) for Prediction and Planning

In contrast to perception, the prediction and planning features of the e2e model do not have a natural two- or three-dimensional spatial structure. These stages feature sequences are of the form $\mathbf{F_i} \in \mathbb{R}^{B\times S}$. We therefore compress respective features into a representation of dimension D that can be aligned with VLM-processed features adopting a stack of L fully-connected (MLP) layers, each followed by Layer Normalization and a ReLU activation.

All linear projection layers are are implemented with normalization and a ReLU nonlinearity as described in Tab. V. Ultimately the output is again projected to the hidden dimension D_{hidden}.

This output $f_{\mathcal{P}}$ serves as the unified feature representation for the downstream prediction and planning module. Note that for prediction and planning stage we will use two separate models, that do not share any parameters as they are handling different tasks respectively.

$\text{PFP}_{\phi}^{\{\texttt{prediction, planning}\}}$

	Layer	Normalization	Activation	Output Dimension
0	Input: $\mathbf{F^i}$	-	-	$\mathbb{R}^{B\times 6\times S}$
1	Linear	LayerNorm	ReLU	$\mathbb{R}^{B\times D_{\text{hidden}}}$
$l=2,$ $\dots L-1$		...		
L	Linear	LayerNorm	ReLU	$\mathbb{R}^{B\times D_{\text{hidden}}}$
$L+1$	Linear	-	-	$\mathbb{R}^{B\times D}$

TABLE V: **Architecture of the PFP for Prediction and Planning**. To achieve alignment between language and prediction or planning features the presented architecture compresses and extracts language space-aligned information from the respective e2e modules.

APPENDIX C
ABLATION LOSS AND DETAILS

A. Module Variations

We generate a smaller dataset for the ablation study to mitigate the time complexity of using the full dataset. For the module variations ablation study, we aim to reduce training time while preserving the full dataset's diversity. This ablation dataset contains trajectories from each scene in the full dataset that begin at t = 0, 10, and 20 and run for 10 time steps. Since most scenes span around 40 time steps, these samples evenly cover the entire dataset with no overlaps. As shown in

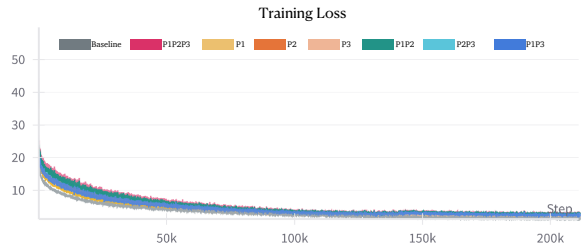


Fig. 8: Training loss over gradient steps for all module variations (each trained for 30 epochs). We observe that as more modules are aligned, the training loss increases. Here, P1, P2, and P3 refer to alignment of the perception, prediction, and planning modules, respectively. The ablation study evaluates different combinations of these alignments, and in every case the loss curves plateau on this small ablation dataset, indicating training saturation.

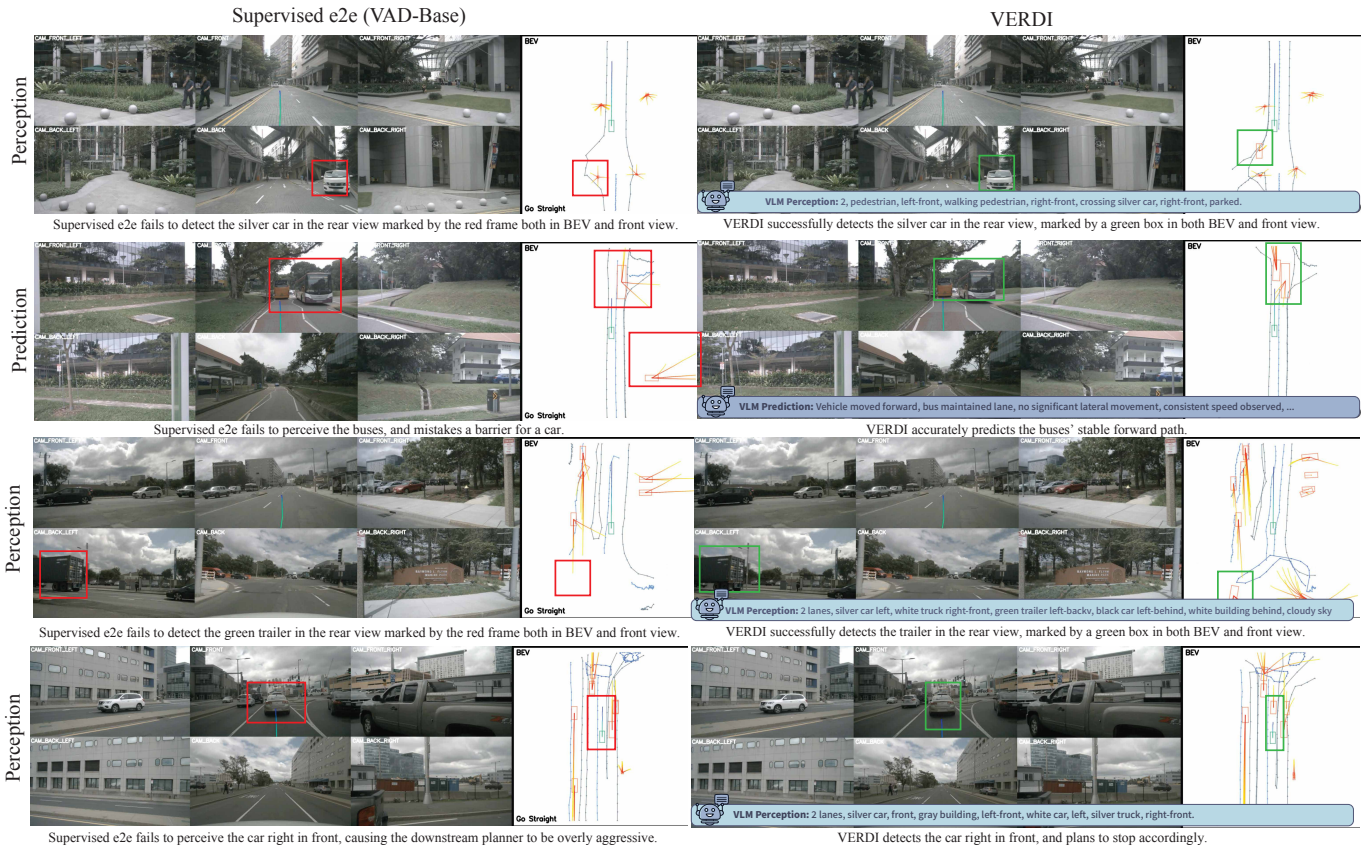
Fig 8, P1, P2 and P3 means perception, prediction and planing alignment separately. All models are trained for 30 epochs and evaluated on the full test set. We show that both VERDI and baseline model trainings saturate on this ablation dataset in Fig 8, making the comparison fair. VERDI shows 8.2% improvement over the baseline method (VAD-tiny), showing the effectiveness of distilling VLM's reasoning capability to the driving modules.

B. VLM Embedding Variations

In this VLM-embedding ablation study, our goal is to reduce training-time complexity while preserving overlap in the training data. Following our module ablation setup, we select three trajectories of ten time steps each. To evaluate the VLM's reasoning consistency, we start those trajectories at t = 0, 5, and 10, which creates five overlapping time steps between each pair. The intuition behind these time step choices is to amplify the impact of embedding quality: a large VLM with strong reasoning capabilities should yield consistent outputs across overlapping time steps—thereby improving performance—whereas a smaller VLM with weaker reasoning will likely produce inconsistent results on the same time steps, degrading performance. The results show that with the Qwen 75B model and the base sentence-transformer encoder, we achieve the largest improvement 6.7% compared to the baseline (VAD-tiny).

APPENDIX D
ADDITIONAL QUALITATIVE RESULTS

We present additional visualization results in Figure 9, illustrating four scenarios in which the supervised e2e model fails—either by missing agents on the road, mispredicting their future trajectories, or planning a path that leads to a collision due to perceptual errors. In contrast, VERDI correctly detects the vehicles behind in the first and third rows (marked by □) and accurately predicts the forward motion of the yellow and white buses (second row, □). Most interestingly, in the fourth row, VERDI comes to a reasonable stop at the red traffic



Supervised e2e fails to detect the silver car in the rear view marked by the red frame both in BEV and front view.

VERDI successfully detects the silver car in the rear view, marked by a green box in both BEV and front view.

Supervised e2e fails to perceive the buses, and mistakes a barrier for a car.

VERDI accurately predicts the buses' stable forward path.

Supervised e2e fails to detect the green trailer in the rear view marked by the red frame both in BEV and front view.

VERDI successfully detects the trailer in the rear view, marked by a green box in both BEV and front view.

Supervised e2e fails to perceive the car right in front, causing the downstream planner to be overly aggressive.

VERDI detects the car right in front, and plans to stop accordingly.

Fig. 9: Additional qualitative comparison of VERDI (Ours, right column) and the Supervised e2e model (baseline, left column) on the nuScenes dataset [13]. Each entry shows the multi-view camera observations on the left and the BEV view on the right at one time step t. The left panel overlays the ego agent's planned 3-second trajectory on the front-camera image and BEV panel as a solid green line that fades to blue. The BEV panel renders the ego vehicle as a green rectangle, pedestrians and other vehicles as red rectangles, and their predicted 3-second trajectories as red lines. Each of the example shows our successful performance on the perception, prediction, and planning modules, indicated by □, while failures are highlighted by □. We also show the VLM texts response for each testing case. We observed that the VERDI's behavior highly aligns with the VLM's response, showing the successful VLM reasoning and commonsense distillation.

light because it perceives the car ahead, whereas the baseline model overlooks this vehicle and proceeds forward, resulting in a collision (marked by □). We also queried the VLM on these cases and observed that VERDI's behavior aligns closely with the VLM's response, demonstrating the successful VLM reasoning distillation.