From Representational Complementarity to Dual Systems: Synergizing VLM and Vision-Only Backbones for End-to-End Driving

Sining Ang[1,2] **Yuguang Yang**[1,3] **Chenxu Dang**[1,4] **Canyu Chen**[1,5] **Cheng Chi**[6]
Haiyan Liu[7] **Xuanyao Mao**[7] **Jason Bao**[7] **Xuliang**[7] **Bingchuan Sun**[7] **Yan Wang**[1*]

[1]Institute for AI Industry Research (AIR), Tsinghua University
[2]University of Science and Technology of China [3]Beihang University
[4]Huazhong University of Science and Technology
[5]National College for Excellent Engineers, Beihang University
[6]School of Information, Renmin University of China
[7]Lenovo Group Limited

Abstract

Vision-Language-Action (VLA) driving augments end-to-end (E2E) planning with language-enabled visual backbones, but how VLM representations differ from vision-only encoders after policy learning, and whether such differences matter for planning, remains unclear. Under a unified VLM-hidden + diffusion-policy paradigm, we compare multiple VLM families/scales (InternVL3 and Qwen3VL) with standard vision-only encoders (ResNet, ViT, and EVA-CLIP) while keeping the downstream planner fixed. We study representation, behavior, and system design. CKA/CCA and Shared–Unique SAE show that policy learning enlarges a common decision subspace, but both branches retain non-transferable residual factors. We further replicate this shared-plus-unique representation pattern on nuPlan using the AsyncDrive planning stack. Latent-intervention policies and scenario-level analysis further show that these residuals are behaviorally meaningful: vision-only encoders are stronger in simple geometry-dominant scenes, whereas VLMs are more effective in semantically complex and interaction-heavy long-tail cases. The two branches also exhibit distinct progress–braking and path-choice tendencies, and an oracle best-of-two VLM+ViT selector reaches 93.58 PDMS on NAVSIM. We convert this complementarity into two lightweight systems: **HybridDriveVLA**, which selects from a compact cross-model candidate set using a learned trajectory scorer and improves PDMS from 90.80 to 92.10, and **DualDriveVLA**, a fast–slow variant that invokes the VLM in only 15% of scenarios, achieving 91.00 PDMS with about $1.9\times$ lower latency than the VLM baseline. Code will be available at `https://github.com/WilliamXuanYu/HybridDriveVLA`.

1 Introduction

End-to-end driving planning maps visual observations directly to future trajectories, reducing reliance on hand-crafted modular pipelines. In this setting, the visual encoder affects not only planning quality but also latency, memory, and deployment cost. Vision–language models (VLMs) are often believed to provide stronger semantics and long-tail reasoning than standard vision-only encoders, yet it remains unclear how the two families differ internally after policy learning, whether the differences are behaviorally meaningful, and how they can be exploited in planning.

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

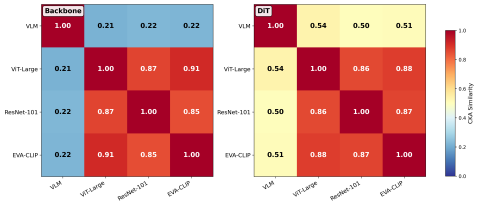


Figure 1: **CKA before and after policy learning.** VLM–vision similarity increases at the decision level without complete collapse.

Table 1: **RQ1 generalization.** The shared-plus-unique trend persists across settings.

Setting	CKA	CCA@10	R^2_{cross}
InternVL3-8B (BB)	0.26/0.23/0.24	0.81	0.49/0.62
InternVL3-8B (Dec.)	0.49/0.48/0.48	0.97	0.54/0.78
Qwen3VL-8B (BB)	0.30/0.29/0.31	0.84	0.44/0.60
Qwen3VL-8B (Dec.)	0.55/0.54/0.54	0.95	0.52/0.81
AsyncDrive/nuPlan	0.66	0.88	0.61/0.73

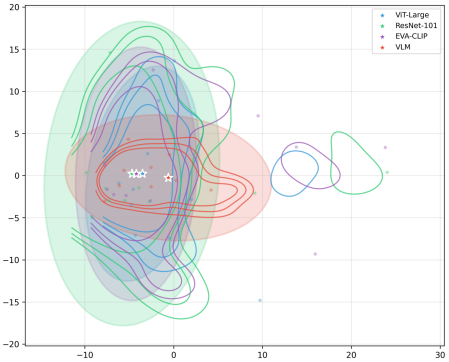


Figure 2: **Aligned feature geometry after Procrustes alignment.** The VLM shares a common core with vision-only encoders while retaining model-specific residual regions.

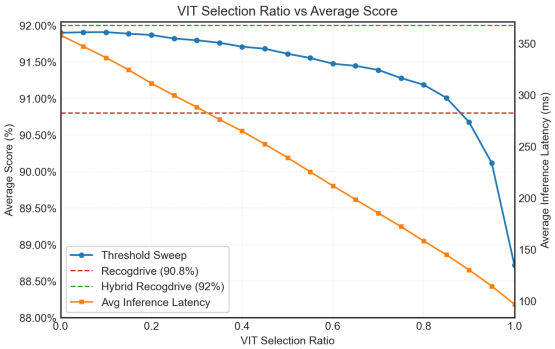


Figure 3: **DualDriveVLA accuracy–compute trade-off.** Varying threshold γ trades off fast-path ratio, PDMS, and inference latency.

We study this question under a unified *VLM-hidden + diffusion-policy* paradigm. Our main experiments instantiate two separate policies with the same downstream diffusion planner and action head: one branch uses a driving-domain-adapted VLM encoder, while the other replaces it with a vision-only encoder. This setup enables controlled comparisons between practical VLA-style and vision-only representation pipelines while keeping the downstream planner, trajectory format, data split, and training schedule fixed. We evaluate multiple VLM families/scales and vision-only encoders on NAVSIM, and include additional cross-setting evidence to test whether the observed patterns persist beyond a single backbone or planner.

We organize the analysis around three questions.

RQ1: representation. How similar are VLM and vision-only representations, and where do the differences remain? Using linear CKA, CCA, and a Shared–Unique SAE, we find that downstream policy learning substantially increases cross-branch alignment: in the main VLM–ViT setting, CKA rises from roughly 0.22 at the backbone level to 0.54 at the decision level. However, the branches do not collapse into a single redundant representation. SAE/CCA analyses consistently reveal a large shared decision subspace together with persistent non-transferable residual factors. Moreover, representation-only gates built from alignment or latent statistics provide only marginal gains over the VLM baseline (best PDMS $90.80 \rightarrow 90.96$), far below the oracle upper bound. This negative result motivates moving from static representation cues to trajectory- and behavior-level reasoning.

RQ2: behavior. Do these residual factors translate into meaningful planning differences? We find stable but long-tailed complementarity rather than simple dominance. At the aggregate level, VLM and vision-only policies exhibit different progress–braking and path-choice tendencies. At the scenario level, vision-only encoders are relatively stronger in simple geometry-dominant scenes, while VLM-specific factors are more useful in semantically complex, ambiguous, and interaction-heavy cases. Under conservative counting, both branches decisively win on persistent subsets of scenarios, and simply choosing the better trajectory between a VLM branch and a ViT branch yields an oracle best-of-two score of 93.58 PDMS on NAVSIM. Latent-intervention policies further support that the SAE-separated residual factors are behaviorally useful rather than merely reconstruction artifacts.

RQ3: system. How can this complementarity be converted into practical accuracy–cost gains? We treat VLM and vision-only policies as complementary candidate generators rather than competitors to be averaged away. *HybridDriveVLA* runs both branches, optionally augments their endpoints with simple interpolated candidates, and uses a learned trajectory scorer for selection, improving PDMS

from 90.80 to 92.10 without changing policy training. *DualDriveVLA* further turns the same idea into a fast–slow deployment policy: it runs the vision-only branch by default and invokes the VLM only when the fast-path score is insufficient. Calling the VLM in only about 15% of scenarios achieves 91.00 PDMS while reducing latency by about $1.9\times$ relative to the VLM baseline.

Overall, our contribution is an analysis-to-mechanism account of VLA driving: policy learning creates substantial shared structure between VLM and vision-only branches, but preserves behaviorally useful residual factors. Rather than distilling these differences away, we show that they can be exploited through lightweight trajectory-level selection and fast–slow deployment.

2 Related Work

VLMs for autonomous driving. Recent work incorporates vision–language models into autonomous driving in two main ways. One line follows a *dual-system* design, where a VLM provides high-level commands, semantic guidance, or intermediate waypoints for a downstream planner (Tian et al., 2024; Jiang et al., 2024). Another line adopts a more *single-system* formulation, casting planning or action generation directly as multimodal language generation, often with prompting or textual reasoning for interpretability (Wang et al., 2024; Mao et al., 2023b; Bai et al., 2024; Shao et al., 2024; Zhang et al., 2024; Mao et al., 2023a; Zhao et al., 2025; Hwang et al., 2024; Wei et al., 2022; Fu et al., 2025; Yang et al., 2026; Chen et al., 2026). We are closer in spirit to the dual-system view, but with a different emphasis: rather than treating the VLM mainly as a stronger semantic module, we ask how VLM and vision-only branches differ after policy learning, and how their residual complementarity can be converted into trajectory selection and efficient deployment.

Diagnosing VLM/VLA driving stacks. Most existing analyses of VLM/VLA driving systems focus on interpretability, reasoning ability, or benchmark construction, such as explanation-centric datasets, language-grounded evaluation, and structured reasoning protocols (Chonghao et al., 2023; Ming et al., 2023; Tianshuai et al., 2025; Sicong et al., 2025; Yue et al., 2025; Zecong et al., 2026). In contrast, we study a more mechanistic question inside a unified planning stack: how VLM and vision-only encoders differ in internal representations, how these differences are transformed—and often compressed—after downstream policy learning, and when the remaining residual factors become behaviorally useful at the scenario level. To our knowledge, this is among the first works to systematically connect *representation geometry*, *behavioral complementarity*, and *trajectory-level selection* within a unified VLA-style planning framework.

Overall, our work complements prior VLM-for-driving systems by offering an analysis-to-mechanism pipeline: from shared-versus-unique representation structure (RQ1), to scenario-level behavioral complementarity (RQ2), to lightweight hybrid/dual designs that exploit this complementarity in practice (RQ3).

3 Preliminaries

3.1 Formulation and Analysis Points

In our NAVSIM experiments, a policy maps visual observation I_{cam}, navigation signal L_{nav}, and ego state S_{ego} to a future trajectory $\hat{\tau} = \Phi(I_{\mathrm{cam}}, L_{\mathrm{nav}}, S_{\mathrm{ego}}) \in \mathbb{R}^{T\times 3}$, where each waypoint contains (x_t, y_t, θ_t).[2]

For RQ1–RQ2, we probe two representations in Fig. 4: the **backbone feature** $\mathbf{h}^{\mathrm{bb}}$ after the encoder adapter, and the **decision feature** $\mathbf{h}^{\mathrm{dec}}$ immediately before the action head. The former isolates encoder-level geometry, while the latter captures what remains after downstream policy learning in the diffusion Transformer planner.

3.2 Unified Two-branch Framework and Comparison Scope

Figure 4 illustrates our two-branch VLM-hidden + diffusion-policy framework. One branch uses a driving-domain-adapted VLM encoder as a frozen source of language-grounded visual tokens,

[2]The notation extends to multi-view inputs by letting $I_{\mathrm{cam}} = \{I^{(v)}\}_{v=1}^{V}$. We use a single front camera.

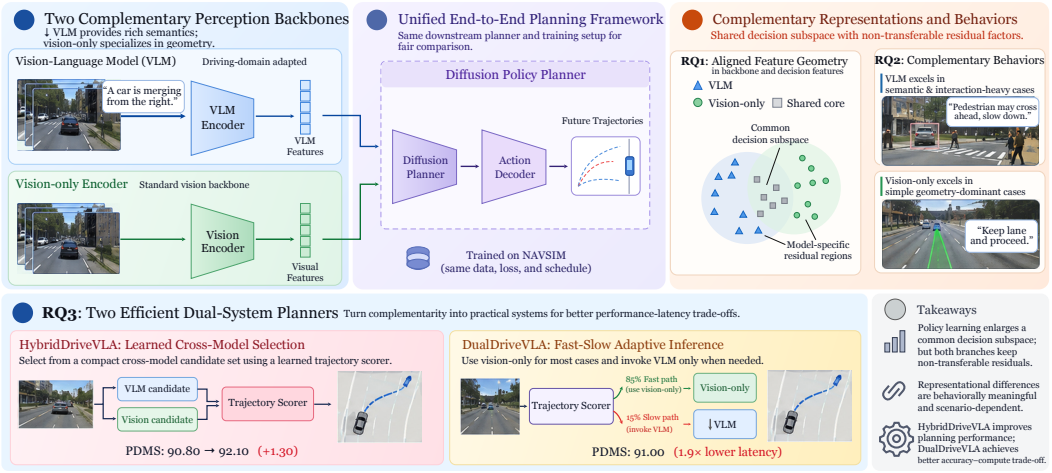


Figure 4: **Overview of our analysis-to-mechanism pipeline.** We compare a VLM branch and a vision-only branch under the same downstream planner/action architecture, probing both backbone and decision features. RQ1 analyzes shared versus branch-specific representation structure; RQ2 connects the residual factors to scenario-level behavioral complementarity; RQ3 exploits this complementarity through trajectory-level selection and fast–slow deployment.

following RecogDrive-style adaptation (Li et al., 2025d); the other replaces it with a standard vision-only encoder such as ViT, ResNet, or EVA-CLIP. Both branches project encoder outputs to the same planner width and use the same DiT planner, action head, trajectory format, data split, and training schedule, but are trained as separate policies without weight sharing. During imitation learning, the driving-domain-adapted VLM encoder remains frozen, whereas the vision-only encoder is trained jointly with the planner and action head. During subsequent reinforcement-learning refinement, both encoders are frozen and only the planner and action head are updated.

Our comparison is therefore between two practical representation pipelines under a controlled downstream planner, rather than a fully symmetric upstream pretraining-budget comparison. This distinction is important because current VLA driving systems commonly adapt a generic VLM to driving data before freezing it as a representation source, whereas vision-only E2E planners typically start from strong visual checkpoints and learn the driving policy through downstream planner training. Thus, differences should be attributed to the representation pipelines made available to the same planner, not to changes in the planner or action head. Full architectural and training details are provided in Appendix A.1.

Domain-adaptation control. To test whether complementarity is mainly due to extra driving-domain exposure in the VLM branch, we add a NAVSIM-restricted control in Appendix F. **VLM-NAVSIM-DA** restricts VLM adaptation to NAVSIM training data and its available text/trajectory annotations. **ViT-NAVSIM-DA** initializes from the same public ViT-large checkpoint, performs MAE adaptation on NAVSIM training images only, discards the MAE decoder, and then follows the original ViT downstream planner training protocol. No NAVSIM test images or evaluation labels are used during adaptation. This control does not make VLM and ViT upstream objectives identical, but tests the narrower confound of whether the VLM–vision complementarity disappears once both branches receive in-domain adaptation on the same dataset.

4 RQ1: Representation Analysis

4.1 Paired Features and Preprocessing

We analyze paired representations extracted from the same driving scenarios. Let $(x_i, y_i)_{i=1}^n$ be aligned feature pairs from the VLM and vision-only branches, stacked as $X, Y \in \mathbb{R}^{n \times d}$. We probe two levels. At the **backbone level** ($d = 384$), the VLM branch produces adapted tokens $F_{\text{vlm}} \in \mathbb{R}^{L \times 384}$, with $L = 2800$ in the main setting; we use $x = \text{MeanPool}(F_{\text{vlm}})$ for tractability, while the vision-only branch provides an adapted global embedding $y \in \mathbb{R}^{384}$.[3] At the **decision level**

[3]Non-pooled analyses on random subsets yield the same qualitative conclusions; details are provided in the appendix E.

($d = 512$), we extract the planner representation after the final denoising step and immediately before the action head for both branches. Unless stated otherwise, features are centered across samples; SAE additionally uses training-split z-score normalization (Appendix A.2).

4.2 Linear Similarity: CKA and CCA

We use two complementary linear tools. Linear CKA measures global geometric similarity, computed on centered features as $\mathrm{CKA}(X, Y) = \|X^\top Y\|_F^2 / (\|X^\top X\|_F \|Y^\top Y\|_F)$, and is reported at both backbone and decision levels to test whether policy learning increases cross-branch alignment. PCA-whitened CCA measures alignability through maximally correlated linear directions. After PCA truncation and whitening for numerical stability, we obtain canonical correlations $\rho_1 \geq \cdots \geq \rho_k$ and summarize them by the spectrum and aggregates such as mean@k; full details are in Appendix A.4.

4.3 Shared–Unique Sparse Autoencoder (SAE)

CKA/CCA quantify similarity but do not separate shared from branch-specific factors, nor test whether the shared component is functionally interchangeable. We therefore train a Shared–Unique SAE on standardized paired features. For each pair (x, y), the SAE encodes shared and unique latents $(z_s^x, z_u^x) = f^x(x)$ and $(z_s^y, z_u^y) = f^y(y)$, and reconstructs with additive linear decoders $\hat{x} = W_s^x z_s^x + W_u^x z_u^x + b^x$ and $\hat{y} = W_s^y z_s^y + W_u^y z_u^y + b^y$. This additive form lets us measure shared and unique contributions in the original feature space. Unless stated otherwise, we use $d_s = 64$ and $d_u = 16$.

Intuitively, the objective seeks a partially interchangeable shared component and compact branch-specific residuals. Full and shared-only reconstruction preserve information; cross shared-only reconstruction (e.g., decoding x from z_s^y) encourages partial interchangeability. VICReg-style alignment, variance, and decorrelation terms align shared codes while discouraging collapse and redundancy (Bardes et al., 2021). Shared–unique orthogonality discourages information leakage between components, and ℓ_1 sparsity promotes compact residual codes. These are established representation-factorization ingredients, with related precedents in Factorized Orthogonal Latent Spaces (Salzmann et al., 2010), Domain Separation Networks (Bousmalis et al., 2016), Learning Factorized Multimodal Representations (Tsai et al., 2018), and MISA (Hazarika et al., 2020); the full objective is in Appendix A.6.

For evaluation, we report explained variance R^2 for full, shared-only self, and shared-only cross reconstruction, and use the self–cross gap $\Delta_{\mathrm{cross}} = R^2_{\mathrm{sh}} - R^2_{\mathrm{cross}}$ as the main interchangeability statistic (Appendix A.7).

4.4 Results & Discussion

Policy learning increases alignment without causing collapse. We first compare the VLM and vision-only branches in the original feature space using linear CKA and PCA-whitened CCA. In the main InternVL3-2B setting, VLM-to-vision CKA increases from about 0.22 at the backbone level to about 0.54 at the decision level (Fig. 1), indicating stronger cross-branch alignment after the shared diffusion planner. CCA shows the same trend from a subspace perspective: although mean canonical correlation changes only mildly ($0.61 \rightarrow 0.63$), the number of high-correlation directions with $\rho > 0.8$ expands from $5/28$ to $28/78$, and the aligned energy increases from $28\% \rightarrow 53\%$ for VLM and $56\% \rightarrow 77\%$ for ViT. Together with the aligned geometry in Fig. 2, this suggests that policy learning compresses heterogeneous encoder evidence into a larger shared decision space, but does not make the two branches redundant.

Shared–Unique SAE as a substitutability diagnostic. CKA/CCA measure alignment, but not whether the shared factors inferred from one branch can substitute for those of the other. We therefore use the Shared–Unique SAE to evaluate partial cross-branch substitutability through cross shared-only reconstruction R^2_{cross} and the self–cross gap Δ_{cross}. Table 2 should be read as a structured diagnostic rather than causal evidence: shared-space alignment is encouraged by the SAE objective and is therefore treated as a sanity check, while our main interpretation relies on original-space CKA/CCA, R^2_{cross}, and Δ_{cross}. Decision-level features are more transferable than backbone features, especially on the ViT side, yet cross reconstruction remains far from perfect and the VLM-side improvement is modest. Thus, downstream policy learning enlarges the shared component while preserving branch-

Table 2: **SAE/CCA summary in the main setting.** R^2_{cross} and Δ_{cross} measure partial substitutability of learned shared factors. Decision features are more aligned and partly more transferable, but cross reconstruction remains imperfect.

Feature	CKA_{orig}	$R^2_{\text{cross}}(x \leftarrow z_s^y / y \leftarrow z_s^x)$	$\Delta_{\text{cross}}(x/y)$	CCA mean@10	CCA AER
Backbone	0.213	0.537/0.623	0.098/0.160	0.800	0.286/0.556
Decision	0.537	0.546/0.763	0.071/0.063	0.972	0.534/0.771

specific residual factors. Their behavioral relevance is tested later through scenario-level analysis and latent-intervention policies.

The shared-plus-unique pattern generalizes beyond the main setting. Table 1 summarizes additional evidence across larger VLMs on NAVSIM and an AsyncDrive/nuPlan cross-stack check. Across InternVL3-8B and Qwen3VL-8B, backbone alignment remains moderate, decision-level alignment increases after policy learning, and SAE cross reconstruction remains non-trivial. The AsyncDrive/nuPlan result shows the same qualitative shared-plus-unique pattern under a different planner/dataset setting. These results may not prove universality, but reduce the concern that the observed pattern is specific to one VLM family or planning stack.

SAE sanity check. Because the SAE objective explicitly encourages shared-latent alignment, shared-space CKA often saturates and is used only as a training-validity check. A shuffled-pair control, which randomly permutes (x_i, y_i) pairings while keeping marginals fixed, substantially reduces shared-space alignment and collapses original-space alignment to near zero (Appendix A.9). We therefore base our main comparisons on original-space alignment and substitutability metrics, namely CKA/CCA, R^2_{cross}, and Δ_{cross}.

4.5 Negative Result: Representation-only Gating is Insufficient

Before turning to RQ2, we test whether one can select between the VLM and ViT trajectories using representation-level signals alone. The gate uses backbone/decision features and SAE-derived shared/unique statistics to predict which branch will obtain the higher closed-loop score. We evaluate handcrafted rules based on shared/unique energy ratios and learned gates including tree-based models, MLPs, and attention-based predictors (Appendix A.11, A.13).

Across settings, representation-only gating does not reliably outperform the VLM baseline and remains far below the oracle best-of-two (Table 5). Even the best learned gate improves PDMS only marginally, from 90.80 to 90.96, whereas the oracle reaches 93.58. Feature visualizations also show poor separation between "VLM-better" and "ViT-better" cases. This negative result motivates RQ2: reliable selection requires trajectory- and behavior-level signals rather than static representation cues alone.

4.6 Takeaways (RQ1)

RQ1 shows that policy learning enlarges the shared VLM–vision subspace, but does not remove branch-specific residuals. Shared–Unique SAE provides a substitutability test showing that decision features are only partially transferable, and the same shared-plus-unique pattern appears across larger VLMs and an extra planner/dataset setting. However, representation-only gating yields only marginal gains over the VLM baseline, motivating the behavioral analysis in RQ2.

5 RQ2: Behavioral Complementarity

5.1 Scenario-level complementarity in the long tail

To characterize behavioral complementarity, we assign each scenario i and policy m an offline trajectory-quality score $s(m, i)$ (NAVSIM PDMS; Appendix B). For a VLM–vision pair, we define the advantage as $\Delta_i(m) = s(\text{VLM}, i) - s(m, i)$ and count a significant win when $|\Delta_i(m)| > \tau$.

A recurring pattern is that complementarity is long-tailed: most scenarios show small score differences, but both branches decisively win on non-trivial and non-nested subsets. Using a stability-aware counting protocol (Appendix C), we observe 257 VLM wins vs. 253 ViT wins at $\tau = 0.2$, and 159

Table 3: **Latent-intervention study.** Policies use different SAE-decomposed decision latents.

Latents used	PDMS	Complex
Shared only	83.0	33 / 187
Shared + ViT-unique	86.1	37 / 187
Shared + VLM-unique	86.9	86 / 187
Shared + both unique	87.5	99 / 187

Table 4: **Behavioral diagnostics on NAVSIM.** Sig. denotes scenarios where the two branches differ substantially in score.

Metric	VLM	ViT	VLM sig.	ViT sig.
Mean speed (m/s)	5.31	5.17	5.74	5.43
Path length (m)	18.60	18.09	20.08	19.02
Min long. acc. (m/s^2)	-2.87	-2.69	-2.99	-2.53
Cum. deceleration	4.94	4.63	5.26	4.38
Hard-brake segs.	0.49	0.44	0.49	0.36

vs. 153 at $\tau = 0.5$. Thus, neither branch strictly contains the other on hard cases, motivating a candidate-generator view rather than a purely average-score comparison.

To identify where the long-tail differences arise, we further use $\tau = 0.9$ to isolate 279 major-decision-mismatch cases and group them into six coarse scene types using the annotation protocol in Appendix G. VLM wins 176 cases and ViT wins 103. The asymmetry becomes clear after semantic grouping: ViT wins more often only in simple lane keeping $(55/92)$, whereas VLM dominates in intersection semantics $(58/78)$, curve/merge/ramp cases $(23/38)$, work zones/cones $(12/15)$, narrow or unclear boundaries $(34/44)$, and occlusion/dense clutter $(12/12)$. Merged into simple vs. complex scenes, VLM wins $139/187$ (74.3%) complex cases, while ViT is stronger in the simple subset.

This breakdown links RQ2 back to RQ1: VLM-specific residual factors are not merely abstract representation leftovers, but become useful in semantic disambiguation, boundary uncertainty, and interaction-heavy scenes. The occlusion/dense-clutter subset is especially indicative, where all 12 cases favor VLM with near-saturated scores while ViT collapses.

5.2 Latent intervention: do unique factors have behavioral utility?

We next test whether SAE-separated unique factors are functional rather than merely reconstructive artifacts. We do not claim strict causal identification, since SAE is not a full causal disentanglement method. Instead, after decomposing decision-level features into shared and branch-specific latents, we train policies with different latent subsets while keeping the downstream architecture and training protocol fixed.

Table 3 shows that shared-only latents already yield 83.0 PDMS, but adding either unique component improves performance. VLM-unique factors provide a larger overall gain than ViT-unique factors (86.9 vs. 86.1), and using both reaches 87.5. More importantly, in the 187 complex scenes, shared+VLM-unique reaches high scores in 86 cases, compared with 37 for shared+ViT-unique and 33 for shared-only; conversely, ViT-unique is more helpful in simple geometry-dominant scenes. Thus, while not a formal causal proof, this intervention-style evidence supports that unique factors have systematic, scene-conditioned behavioral utility.

5.3 Behavior differences beyond scenario counts

Complementarity also appears in how the two branches trade off progress, braking, and path choice. Table 4 shows that, relative to ViT, the VLM branch has slightly higher progress-related quantities on the full test set, including mean speed and path length, and stronger braking-side responses, including more negative minimum longitudinal acceleration, larger cumulative deceleration, and more hard-brake segments. These gaps become larger on the significant-difference subset, suggesting that VLM tends to advance further while applying stronger corrections when needed, rather than simply driving uniformly faster.

Beyond longitudinal statistics, we also observe lateral differences in lane-centering, merge timing, and route selection. In many cases, the expert trajectory lies between the two branches or follows an intermediate path, motivating the interpolation-based candidate construction in Sec. 6. Appendix I reports qualitative examples, the full behavioral and safety diagnostics with standard deviations over five random seeds, showing that the trends are stable across runs.

5.4 Implications: selection needs trajectory-level signals

If complementarity is real, selecting the better trajectory from a combined candidate set should yield a meaningful upper bound. For each scenario with candidate set $\mathcal{C}_i$, we define the Best-of-n score as

Table 5: Representation-only gating on NAVSIM. Higher is better; oracle = per-scenario best branch.

Method	PDMS(%)↑
InternVL3-2B	90.80
ViT-base/large	85.62/88.88
Resnet-50/101	86.02/87.69
Evaclip02-base	87.89
Oracle best-of-two (VLM+ViT-large)	**93.58**
Rule: More-unique wins	89.95
Rule: Shared-dominant conditional	90.22
Rule: Smoothed shared-dominance	**90.29**
Rule: ViT-prior fallback	89.92
Random Forest	90.87
Gradient Boosting	90.75
GBDT	90.65
MLP classifier	90.80
Self-attention (binary/regression)	90.82/**90.96**
Self-attention (partial score terms)	90.82

Table 6: **Trajectory-level selection and candidate augmentation on NAVSIM (*navtest*).** Endpoint-only scorer selection already captures much of the cross-model complementarity; interpolation provides an additional lightweight candidate-set augmentation. DualDriveVLA uses the fast branch by default (Sec. 6.2).

Method	PDMS(%)↑
InternVL3-2B	90.80
ViT-large	88.88
Best-of-n (ViT, n=1/3/6)	88.88/89.13/89.32
Best-of-n (VLM, n=1/3/6)	90.80/91.57/91.95
Cross-model oracle (Best-of-2)	93.58
Cross-model oracle (Best-of-6)	94.00
Trajectory mean (two endpoints)	91.18
Rule-based selection (grid search)	91.21
Adaptive weighting (predict α)	91.31
Scorer selection (endpoints only)	91.75
HybridDriveVLA	**92.10**

$S_{\mathrm{BoN}}(i) = \max_{\tau \in \mathcal{C}_i} s(\tau, i)$. With $\mathcal{C}_i = \{\tau_{\mathrm{VLM}}(i), \tau_{\mathrm{ViT}}(i)\}$, Best-of-2 improves PDMS from 90.80 to 93.58 (Table 6), confirming set-level complementarity.

Together with RQ1, these results show that reliable selection should rely on trajectory-/behavior-level signals rather than static representation cues alone. A practical implication is to score a small set of complementary candidates—the VLM and ViT trajectories and their interpolations—to recover part of the long-tail Best-of-n gain without relying on large within-model sampling.

6 RQ3: From Behavioral Complementarity to Trajectory Selection

RQ1 shows that representation-level cues are insufficient for reliable sample-wise selection, while RQ2 shows stable trajectory-level complementarity between VLM and vision-only policies. RQ3 therefore turns this complementarity into a practical mechanism: construct a small cross-model candidate set and select using trajectory-level scores. This is motivated by the limited diversity of within-model diffusion sampling, where increasing samples from a single ViT or VLM gives only marginal Best-of-n gains (Table 6), consistent with prior observations for RecogDrive-style planners (Chen et al., 2026). We evaluate on NAVSIM *navtest* using its pseudo closed-loop metrics: PDMS for NAVSIM-v1 and EPDMS for NAVSIM-v2 (Appendix B). Tables 7 and 8 compare against prior and concurrent methods, complementing the ablations in Table 6.

6.1 HybridDriveVLA: cross-model candidates and trajectory scoring

For each scenario, HybridDriveVLA starts from the two endpoint trajectories τ_{vit} and τ_{vlm}. Motivated by RQ2, where the two branches often differ in progress–braking–path-choice trade-offs and expert-like trajectories can lie between them, we augment the endpoints with a simple interpolation set: $\tau_\alpha = \alpha\tau_{\mathrm{vit}} + (1-\alpha)\tau_{\mathrm{vlm}}$, with $\alpha \in \{0.1, \ldots, 0.9\}$. The final candidate set is $\mathcal{C} = \{\tau_{\mathrm{vit}}, \tau_{\mathrm{vlm}}, \tau_{0.1}, \ldots, \tau_{0.9}\}$. This interpolation is deliberately modest: it is not a general trajectory generator, but a low-cost local expansion around two complementary endpoints. Table 6 shows that endpoint-only scorer selection already captures much of the complementarity, while interpolation provides an additional gain.

Given $\mathcal{C}$, we train a lightweight trajectory scorer to predict PDMS-related sub-score components, following the spirit of DrivoR-style scoring (Kirby et al., 2026). The scorer encodes the current scene with DINOv2-Small and embeds each candidate trajectory τ into a query $q_\tau = f_{\mathrm{wp}}(\tau) \in \mathbb{R}^{D_{\mathrm{score}}}$. Trajectory-conditioned heads then predict safety-, progress-, and comfort-related components, supervised by the oracle evaluator with component-wise losses. At inference, the predicted components are composed into a meta-score $\hat{s}(\tau)$ following the PDMS structure, and the selected trajectory is $\tau^\star = \arg\max_{\tau \in \mathcal{C}} \hat{s}(\tau)$. The scorer does not generate trajectories; it only ranks candidates from the complementary VLM and vision-only branches.

Table 7: **Performance comparison on NAVSIM-v1 *navtest* using PDMS.**

Method	NC↑	DAC↑	TTC↑	Comf.↑	EP↑	PDMS↑
Traditional E2E planners						
UniAD (Hu et al., 2023)	97.8	91.9	92.9	**100**	78.8	83.4
Hydra-MDP (Li et al., 2024)	98.3	96.0	94.6	**100**	78.7	86.5
DiffusionDrive (Liao et al., 2024)	98.2	96.2	94.7	**100**	82.2	88.1
WoTE (Li et al., 2025c)	98.5	96.8	94.9	99.9	81.9	88.3
DiffusionDriveV2 (Zou et al., 2025)	98.3	97.9	94.8	99.9	**88.0**	91.2
iPad (Guo et al., 2025)	98.6	98.3	94.9	**100**	**88.0**	91.7
VLA-based planners						
DrivingGPT (Chen et al., 2024b)	98.9	90.7	94.9	95.6	79.7	82.4
PARA-Drive (Weng et al., 2024)	97.9	92.4	93.0	99.8	79.3	84.0
DRAMA (Yuan et al., 2024)	98.0	93.1	94.8	100	80.1	85.5
ImagiDrive (Li et al., 2025a)	98.1	96.2	94.5	100	80.5	86.9
AutoVLA (Li et al., 2025b)	98.4	95.6	**98.0**	99.9	81.9	89.1
DriveVLA-W0 (Zhou et al., 2025)	98.7	99.1	95.3	99.3	83.3	90.2
Curious-VLA (Chen et al., 2026)	98.4	96.9	97.9	98.1	88.5	90.2
ReCogDrive (Li et al., 2025d)	97.9	97.3	94.9	**100**	87.3	90.8
WAM-diff (Xu et al., 2025)	**99.1**	98.3	96.5	99.9	84.4	91.0
HybridDriveVLA(ours)	98.6	**98.6**	96.2	**100**	87.3	**92.1**

Table 8: **Performance comparison on NAVSIM-v2 *navtest* using EPDMS.**

Method	NC↑	DAC↑	EP↑	TTC↑	C↑	TL↑	DDC↑	LK↑	EC↑	EPDMS↑
Transfuser (Chitta et al., 2022)	97.7	92.8	79.2	92.8	**100**	99.9	98.3	67.6	95.3	77.8
VADv2 (Chen et al., 2024a)	97.3	91.7	77.6	92.7	**100**	99.9	98.2	66.0	97.4	76.6
Hydra-MDP++ (Li et al., 2024)	97.9	96.5	79.2	93.4	**100**	**100.0**	98.9	67.2	97.7	80.6
ARTEMIS (Feng et al., 2025)	98.3	95.1	81.5	97.4	**100**	99.8	98.6	96.5	**98.3**	83.1
ReCogDrive (Li et al., 2025d)	98.3	95.2	87.1	97.5	98.3	99.8	**99.5**	**96.6**	86.5	83.6
DiffusionDriveV2 (Zou et al., 2025)	97.7	**96.6**	88.9	97.2	97.8	99.8	99.2	96.0	91.0	85.5
HybridDriveVLA(ours)	**98.6**	92.2	**89.7**	**98.5**	98.3	99.8	99.3	**96.6**	87.0	**85.5**

6.2 DualDriveVLA: fast–slow deployment

HybridDriveVLA improves accuracy by always running both branches. DualDriveVLA converts the same complementarity into a fast–slow policy: it first runs the ViT branch and scores τ_{vit}; if $\hat{s}(\tau_{\text{vit}}) \geq \gamma$, it directly outputs the fast-path trajectory, otherwise it invokes the VLM branch, constructs the hybrid candidate set, and applies the scorer selection. Sweeping γ exposes an accuracy–compute trade-off (Fig. 3): lower thresholds favor the fast path, while higher thresholds approach HybridDriveVLA by invoking the VLM more often. On a single NVIDIA A30 GPU, the VLM baseline obtains 90.80 PDMS with about 280 ms latency. With the recommended threshold, DualDriveVLA invokes the VLM in only about 15% of scenarios, achieves 91.00 PDMS, and reduces latency to about 150 ms, corresponding to a $1.9\times$ speedup while slightly improving accuracy.

7 Limitations

Our analysis focuses on VLA driving systems that use a native VLM encoder as a source of language-grounded visual hidden states for an end-to-end planner. This enables comparison with vision-only encoders under the same downstream planner, isolating the representation and behavior differences introduced by the encoder pipeline.

The analysis is less directly applicable to LLM-centric architectures such as Orion (Fu et al., 2025) or SimLingo (Renz et al., 2025), which inject tokens from an external visual front-end into a primarily text-based LLM. Without a standalone native VLM encoder, applying our shared–unique analysis requires redefining the comparison unit, for example by comparing visual front-ends before LLM fusion or probing representations before and after multimodal token fusion.

8 Conclusion

We studied whether VLM and vision-only backbones remain redundant or complementary after policy learning in end-to-end driving. Across representation, behavior, and system analyses, we find that downstream policy learning enlarges a shared decision subspace but preserves branch-specific residual factors that lead to long-tailed behavioral complementarity. Static representation-based

gates cannot reliably exploit this complementarity, whereas trajectory-level selection over a small cross-model candidate set yields stable gains. HybridDriveVLA improves NAVSIM PDMS from 90.80 to 92.10, and DualDriveVLA retains 91.00 PDMS while invoking the VLM in only 15% of scenarios, reducing latency from about 280 ms to 150 ms on a single NVIDIA A30 GPU.

Acknowledgments and Disclosure of Funding

Funded by Xiongan AI Institute, Lenovo Research and Wuxi Research Institute of Applied Technologies, Tsinghua University under Grant 20242001120.

References

Yifan Bai, Dongming Wu, Yingfei Liu, Fan Jia, Weixin Mao, Ziheng Zhang, Yucheng Zhao, Jianbing Shen, Xing Wei, Tiancai Wang, et al. Is a 3d-tokenized llm the key to reliable autonomous driving? *arXiv preprint arXiv:2405.18361*, 2024.

Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. *arXiv preprint arXiv:2105.04906*, 2021.

Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. Domain separation networks. *Advances in neural information processing systems*, 29, 2016.

Canyu Chen, Yuguang Yang, Zhewen Tan, Yizhi Wang, Ruiyi Zhan, Haiyan Liu, Xuanyao Mao, Jason Bao, Xinyue Tang, Linlin Yang, et al. Devil is in narrow policy: Unleashing exploration in driving vla models. *arXiv preprint arXiv:2603.06049*, 2026.

Shaoyu Chen, Bo Jiang, Hao Gao, Bencheng Liao, Qing Xu, Qian Zhang, Chang Huang, Wenyu Liu, and Xinggang Wang. Vadv2: End-to-end vectorized autonomous driving via probabilistic planning. *arXiv preprint arXiv:2402.13243*, 2024a.

Yuntao Chen, Yuqi Wang, and Zhaoxiang Zhang. Drivinggpt: Unifying driving world modeling and planning with multi-modal autoregressive transformers. *arXiv preprint arXiv:2412.18607*, 2024b.

Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger. Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(11):12878–12895, 2022.

Sima Chonghao, Renz Katrin, Chitta Kashyap, Chen Li, Zhang Hanxue, Xie Chengen, Beißwenger Jens, Luo Ping, Geiger Andreas, and Li Hongyang. Drivelm: Driving with graph visual question answering. *arXiv preprint arXiv:2312.14150v3*, 2023. URL `https://www.arxiv.org/abs/2312.14150v3`.

Renju Feng, Ning Xi, Duanfeng Chu, Rukang Wang, Zejian Deng, Anzheng Wang, Liping Lu, Jinxiang Wang, and Yanjun Huang. Artemis: Autoregressive end-to-end trajectory planning with mixture of experts for autonomous driving. *arXiv preprint arXiv:2504.19580*, 2025.

Haoyu Fu, Diankun Zhang, Zongchuang Zhao, Jianfeng Cui, Dingkang Liang, Chong Zhang, Dingyuan Zhang, Hongwei Xie, Bing Wang, and Xiang Bai. Orion: A holistic end-to-end autonomous driving framework by vision-language instructed action generation. *arXiv preprint arXiv:2503.19755*, 2025.

Ke Guo, Haochen Liu, Xiaojun Wu, Jia Pan, and Chen Lv. ipad: Iterative proposal-centric end-to-end autonomous driving. *arXiv preprint arXiv:2505.15111*, 2025.

Devamanyu Hazarika, Roger Zimmermann, and Soujanya Poria. Misa: Modality-invariant and-specific representations for multimodal sentiment analysis. In *Proceedings of the 28th ACM international conference on multimedia*, pages 1122–1131, 2020.

Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, et al. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 17853–17862, 2023.

Jyh-Jing Hwang, Runsheng Xu, Hubert Lin, Wei-Chih Hung, Jingwei Ji, Kristy Choi, Di Huang, Tong He, Paul Covington, Benjamin Sapp, et al. Emma: End-to-end multimodal model for autonomous driving. *arXiv preprint arXiv:2410.23262*, 2024.

Bo Jiang, Shaoyu Chen, Bencheng Liao, Xingyu Zhang, Wei Yin, Qian Zhang, Chang Huang, Wenyu Liu, and Xinggang Wang. Senna: Bridging large vision-language models and end-to-end autonomous driving. *arXiv preprint arXiv:2410.22313*, 2024.

Ellington Kirby, Alexandre Boulch, Yihong Xu, Yuan Yin, Gilles Puy, Éloi Zablocki, Andrei Bursuc, Spyros Gidaris, Renaud Marlet, Florent Bartoccioni, et al. Driving on registers. *arXiv preprint arXiv:2601.05083*, 2026.

Jingyu Li, Bozhou Zhang, Xin Jin, Jiankang Deng, Xiatian Zhu, and Li Zhang. Imagidrive: A unified imagination-and-planning framework for autonomous driving. *arXiv preprint arXiv:2508.11428*, 2025a.

Yingyan Li, Shuyao Shang, Weisong Liu, Bing Zhan, Haochen Wang, Yuqi Wang, Yuntao Chen, Xiaoman Wang, Yasong An, Chufeng Tang, et al. Drivevla-w0: World models amplify data scaling law in autonomous driving. *arXiv preprint arXiv:2510.12796*, 2025b.

Yingyan Li, Yuqi Wang, Yang Liu, Jiawei He, Lue Fan, and Zhaoxiang Zhang. End-to-end driving with online trajectory evaluation via bev world model. *arXiv preprint arXiv:2504.01941*, 2025c.

Yongkang Li, Kaixin Xiong, Xiangyu Guo, Fang Li, Sixu Yan, Gangwei Xu, Lijun Zhou, Long Chen, Haiyang Sun, Bing Wang, et al. Recogdrive: A reinforced cognitive framework for end-to-end autonomous driving. *arXiv preprint arXiv:2506.08052*, 2025d.

Zhenxin Li, Kailin Li, Shihao Wang, Shiyi Lan, Zhiding Yu, Yishen Ji, Zhiqi Li, Ziyue Zhu, Jan Kautz, Zuxuan Wu, et al. Hydra-mdp: End-to-end multimodal planning with multi-target hydra-distillation. *arXiv preprint arXiv:2406.06978*, 2024.

Bencheng Liao, Shaoyu Chen, Haoran Yin, Bo Jiang, Cheng Wang, Sixu Yan, Xinbang Zhang, Xiangyu Li, Ying Zhang, Qian Zhang, et al. Diffusiondrive: Truncated diffusion model for end-to-end autonomous driving. *arXiv preprint arXiv:2411.15139*, 2024.

Jiageng Mao, Yuxi Qian, Junjie Ye, Hang Zhao, and Yue Wang. Gpt-driver: Learning to drive with gpt. *arXiv preprint arXiv:2310.01415*, 2023a.

Jiageng Mao, Junjie Ye, Yuxi Qian, Marco Pavone, and Yue Wang. A language agent for autonomous driving. *arXiv preprint arXiv:2311.10813*, 2023b.

Nie Ming, Peng Renyuan, Wang Chunwei, Cai Xinyue, Han Jianhua, Xu Hang, and Zhang Li. Reason2drive: Towards interpretable and chain-based reasoning for autonomous driving. *arXiv preprint arXiv:2312.03661*, 2023. URL `https://www.arxiv.org/abs/2312.03661`.

Katrin Renz, Long Chen, Elahe Arani, and Oleg Sinavski. Simlingo: Vision-only closed-loop autonomous driving with language-action alignment. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 11993–12003, 2025.

Mathieu Salzmann, Carl Henrik Ek, Raquel Urtasun, and Trevor Darrell. Factorized orthogonal latent spaces. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 701–708. JMLR Workshop and Conference Proceedings, 2010.

Hao Shao, Yuxuan Hu, Letian Wang, Guanglu Song, Steven L Waslander, Yu Liu, and Hongsheng Li. Lmdrive: Closed-loop end-to-end driving with large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15120–15130, 2024.

Jiang Sicong, Huang Zilin, Qian Kangan, Luo Ziang, Zhu Tianze, Zhong Yang, Tang Yihong, Kong Menglin, Wang Yunlong, Jiao Siwen, Ye Hao, Sheng Zihao, Zhao Xin, Wen Tuopu, Fu Zheng, Chen Sikai, Jiang Kun, Yang Diange, Choi Seongjin, and Sun Lijun. A survey on vision-language-action models for autonomous driving. *arXiv preprint arXiv:2506.24044*, 2025. URL `https://www.arxiv.org/abs/2506.24044`.

Xiaoyu Tian, Junru Gu, Bailin Li, Yicheng Liu, Yang Wang, Zhiyong Zhao, Kun Zhan, Peng Jia, Xianpeng Lang, and Hang Zhao. Drivevlm: The convergence of autonomous driving and large vision-language models. *arXiv preprint arXiv:2402.12289*, 2024.

Hu Tianshuai, Liu Xiaolu, Wang Song, Zhu Yiyao, Liang Ao, Kong Lingdong, Zhao Guoyang, Gong Zeying, Cen Jun, Huang Zhiyu, Hao Xiaoshuai, Li Linfeng, Song Hang, Li Xiangtai, Ma Jun, Shen Shaojie, Zhu Jianke, Tao Dacheng, Liu Ziwei, and Liang Junwei. Vision-language-action models for autonomous driving: Past, present, and future. *arXiv preprint arXiv:2512.16760v2*, 2025. URL `https://www.arxiv.org/abs/2512.16760v2`.

Yao-Hung Hubert Tsai, Paul Pu Liang, Amir Zadeh, Louis-Philippe Morency, and Ruslan Salakhutdinov. Learning factorized multimodal representations. *arXiv preprint arXiv:1806.06176*, 2018.

Shihao Wang, Zhiding Yu, Xiaohui Jiang, Shiyi Lan, Min Shi, Nadine Chang, Jan Kautz, Ying Li, and Jose M Alvarez. Omnidrive: A holistic llm-agent framework for autonomous driving with 3d perception, reasoning and planning. *arXiv preprint arXiv:2405.01533*, 2024.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Xinshuo Weng, Boris Ivanovic, Yan Wang, Yue Wang, and Marco Pavone. Para-drive: Parallelized architecture for real-time autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15449–15458, 2024.

Mingwang Xu, Jiahao Cui, Feipeng Cai, Hanlin Shang, Zhihao Zhu, Shan Luan, Yifang Xu, Neng Zhang, Yaoyi Li, Jia Cai, et al. Wam-diff: A masked diffusion vla framework with moe and online reinforcement learning for autonomous driving. *arXiv preprint arXiv:2512.11872*, 2025.

Yuguang Yang, Canyu Chen, Zhewen Tan, Yizhi Wang, Zichao Feng, Chunyang Liu, Kehua Sheng, Juan Zhang, Linlin Yang, Baochang Zhang, et al. Teaching vision-language-action models what to see and where to look. In *European Conference on Computer Vision*, pages 317–335. Springer, 2026.

Chengran Yuan, Zhanqi Zhang, Jiawei Sun, Shuo Sun, Zefan Huang, Christina Dao Wen Lee, Dongen Li, Yuhang Han, Anthony Wong, Keng Peng Tee, et al. Drama: An efficient end-to-end motion planner for autonomous driving with mamba. *arXiv preprint arXiv:2408.03601*, 2024.

Li Yue, Tian Meng, Lin Zhenyu, Zhu Jiangtong, Zhu Dechang, Liu Haiqiang, Wang Zining, Zhang Yueyi, Xiong Zhiwei, and Zhao Xinhai. Fine-grained evaluation of large vision-language models in autonomous driving. *arXiv preprint arXiv:2503.21505v1*, 2025. URL `https://www.arxiv.org/abs/2503.21505v1`.

Tang Zecong, Wang Zixu, Wang Yifei, Lian Weitong, Gao Tianjian, Li Haoran, Ru Tengju, Meng Lingyi, Cui Zhejun, Zhu Yichen, Kang Qi, Wang Kaixuan, and Zhang Yu. Autodridm: An explainable benchmark for decision-making of vision-language models in autonomous driving. *arXiv preprint arXiv:2601.14702v1*, 2026. URL `https://www.arxiv.org/abs/2601.14702v1`.

Songyan Zhang, Wenhui Huang, Zihui Gao, Hao Chen, and Chen Lv. Wisead: Knowledge augmented end-to-end autonomous driving with vision-language model. *arXiv preprint arXiv:2412.09951*, 2024.

Rui Zhao, Qirui Yuan, Jinyu Li, Haofeng Hu, Yun Li, Chengyuan Zheng, and Fei Gao. Sce2drivex: A generalized mllm framework for scene-to-drive learning. *arXiv preprint arXiv:2502.14917*, 2025.

Zewei Zhou, Tianhui Cai, Seth Z Zhao, Yun Zhang, Zhiyu Huang, Bolei Zhou, and Jiaqi Ma. Autovla: A vision-language-action model for end-to-end autonomous driving with adaptive reasoning and reinforcement fine-tuning. *arXiv preprint arXiv:2506.13757*, 2025.

Jialv Zou, Shaoyu Chen, Bencheng Liao, Zhiyu Zheng, Yuehao Song, Lefei Zhang, Qian Zhang, Wenyu Liu, and Xinggang Wang. Diffusiondrivev2: Reinforcement learning-constrained truncated diffusion modeling in end-to-end autonomous driving. *arXiv preprint arXiv:2512.07745*, 2025.

A Detailed Formulas for RQ1

This appendix specifies the full definitions and implementation details omitted from the main text of RQ1, including preprocessing, linear CKA/CCA (with PCA truncation and whitening), and the Shared–Unique SAE objective and metrics.

A.1 RecogDrive Details and Training Protocol

A.1.1 Architecture and Tensor Shapes

VLM branch (InternVL3-2B). Given I_{cam} and a fixed textual prompt, the VLM produces last-layer hidden states

$$H_{\mathrm{vlm}} \in \mathbb{R}^{L \times d_{\mathrm{vlm}}}.$$

A linear adapter maps tokens to the planner width d:

$$F_{\mathrm{vlm}} = H_{\mathrm{vlm}} W_{\mathrm{vlm}}, \qquad F_{\mathrm{vlm}} \in \mathbb{R}^{L \times d}.$$

For feature analysis, the backbone feature is defined by mean pooling:

$$\mathbf{h}_{\mathrm{vlm}}^{\mathrm{bb}} = \mathrm{MeanPool}(F_{\mathrm{vlm}}) \in \mathbb{R}^{d}.$$

Vision-only branch (ViT/ResNet/EVA-CLIP). Each vision-only backbone outputs a global embedding

$$\tilde{\mathbf{h}}_{\mathrm{vis}}^{\mathrm{bb}} \in \mathbb{R}^{d_{\mathrm{vis}}},$$

which is mapped to the same planner width:

$$\mathbf{h}_{\mathrm{vis}}^{\mathrm{bb}} = \tilde{\mathbf{h}}_{\mathrm{vis}}^{\mathrm{bb}} W_{\mathrm{vis}} \in \mathbb{R}^{d}.$$

For interface consistency, $\mathbf{h}_{\mathrm{vis}}^{\mathrm{bb}}$ can be implemented as a length-1 token sequence when the planner expects token inputs.

Diffusion planner and decision feature. A diffusion Transformer planner (DiT) iteratively denoises to produce a latent trajectory representation. The *decision feature* is the planner output immediately before the action head:

$$\mathbf{h}^{\mathrm{dec}} \in \mathbb{R}^{d_{\mathrm{dec}}}.$$

A lightweight MLP action head maps it to $\hat{\tau} \in \mathbb{R}^{T \times 3}$.

Default dimensions. Unless stated otherwise, we use d=384, d_{dec}=512, and T=8.

A.1.2 Training Protocol and Fairness Controls

All variants share the same NAVSIM configuration (split, optimizer, schedule, epochs) and identical planner/action-head architecture.

VLM branch. We follow the RecogDrive recipe: (1) domain adaptation of the VLM on refined image–text driving data with trajectory supervision; (2) imitation learning on NAVSIM with the VLM backbone frozen and the planner/action head trained; (3) reinforcement learning on NAVSIM with the VLM still frozen and only the planner/action head updated.

Vision-only branch. Vision-only backbones are initialized from public pretrained weights (without the image–text domain-adaptation stage). On NAVSIM, we train the vision backbone jointly with the planner/action head during imitation learning. During RL refinement, the vision backbone is frozen and only the planner/action head is updated, matching the VLM branch to isolate backbone effects.

A.1.3 Compute resources.

All main training experiments were conducted on a server with 8 NVIDIA A100 GPUs. The VLM driving-domain adaptation stage takes approximately 2–3 days depending on the VLM scale, with the 8B variants being more expensive than the 3B variants. After obtaining the adapted/frozen encoder, the downstream planner training requires about 20 hours for the imitation-learning stage and about 20 hours for the subsequent reinforcement/refinement stage. The reported latency measurements are obtained on a single NVIDIA A30 GPU, as described in Sec. 6.

A.2 Paired Features, Centering, and Standardization

Paired features. Let $(x_i, y_i)_{i=1}^n$ be aligned feature pairs extracted from the same driving scenarios, and stack them as

$$X = [x_1^\top; \ldots; x_n^\top] \in \mathbb{R}^{n \times d}, \qquad Y = [y_1^\top; \ldots; y_n^\top] \in \mathbb{R}^{n \times d}. \tag{1}$$

At the backbone level, x_i is obtained by mean-pooling the VLM token sequence after the RecogDrive adapter (to support dataset-level statistics and cross-backbone comparability), while y_i is the vision-only global embedding after its adapter (no pooling).

Centering across samples. We center each feature dimension across samples using

$$H = I - \frac{1}{n}\mathbf{1}\mathbf{1}^\top \in \mathbb{R}^{n \times n}, \qquad \tilde{X} = HX,\ \tilde{Y} = HY. \tag{2}$$

Unless stated otherwise, CKA/CCA use centered features.

Per-dimension z-scoring (used for SAE and reported R^2). For SAE training and all reported R^2 values, we standardize each dimension using *training-split, dataset-level* statistics:

$$x' = (x - \mu_x) \oslash \sigma_x, \qquad y' = (y - \mu_y) \oslash \sigma_y, \tag{3}$$

where $\mu_x, \sigma_x \in \mathbb{R}^d$ are computed per-dimension over the training split (and then reused for validation/test). For notational simplicity we drop primes in the SAE sections. (For some visualizations only, we may apply plot-specific normalization; such choices do not affect the quantitative RQ1 results.)

A.3 Linear Similarity: Linear CKA

Given centered matrices $\tilde{X}, \tilde{Y}$, linear CKA is

$$\mathrm{CKA}(X, Y) = \frac{\|\tilde{X}^\top \tilde{Y}\|_F^2}{\|\tilde{X}^\top \tilde{X}\|_F \, \|\tilde{Y}^\top \tilde{Y}\|_F}. \tag{4}$$

A.4 PCA-Truncation and Whitening for CCA

Given centered feature matrices $\tilde{X} \in \mathbb{R}^{n \times d_x}$ and $\tilde{Y} \in \mathbb{R}^{n \times d_y}$, define sample covariances

$$\Sigma_{xx} = \frac{1}{n-1}\tilde{X}^\top \tilde{X}, \qquad \Sigma_{yy} = \frac{1}{n-1}\tilde{Y}^\top \tilde{Y}, \qquad \Sigma_{xy} = \frac{1}{n-1}\tilde{X}^\top \tilde{Y}. \tag{5}$$

PCA truncation (explained-variance threshold). We eigendecompose

$$\Sigma_{xx} = P_x \Lambda_x P_x^\top, \qquad \Sigma_{yy} = P_y \Lambda_y P_y^\top, \tag{6}$$

where $\Lambda_x = \mathrm{diag}(\lambda_1^x, \ldots, \lambda_{d_x}^x)$ and $\lambda_1^x \geq \cdots \geq 0$ (similarly for y). We choose the smallest k_x (resp. k_y) such that the cumulative explained variance reaches $\eta = 0.99$:

$$k_x = \min\left\{ k : \frac{\sum_{j=1}^k \lambda_j^x}{\sum_{j=1}^{d_x} \lambda_j^x} \geq \eta \right\}, \qquad k_y = \min\left\{ k : \frac{\sum_{j=1}^k \lambda_j^y}{\sum_{j=1}^{d_y} \lambda_j^y} \geq \eta \right\}. \tag{7}$$

We then keep

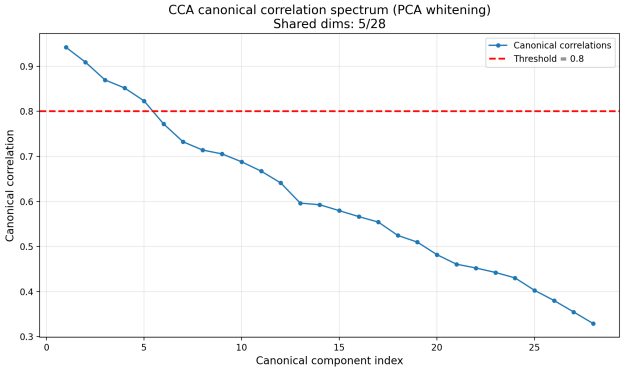
$$P_x^{(k_x)} = P_x[:, 1{:}k_x],\ \Lambda_x^{(k_x)} = \Lambda_x[1{:}k_x, 1{:}k_x], \qquad P_y^{(k_y)} = P_y[:, 1{:}k_y],\ \Lambda_y^{(k_y)} = \Lambda_y[1{:}k_y, 1{:}k_y]. \tag{8}$$

Whitening (ridge-stabilized). We use ridge $\epsilon = 10^{-8}$:

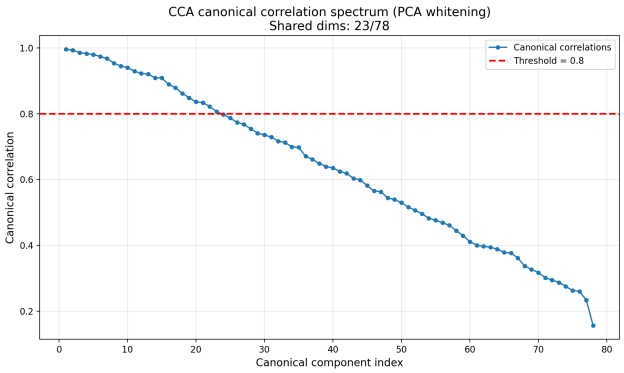
$$W_x = P_x^{(k_x)} \left(\Lambda_x^{(k_x)} + \epsilon I\right)^{-1/2}, \qquad W_y = P_y^{(k_y)} \left(\Lambda_y^{(k_y)} + \epsilon I\right)^{-1/2}, \tag{9}$$

and compute whitened features

$$\hat{X} = \tilde{X} W_x, \qquad \hat{Y} = \tilde{Y} W_y. \tag{10}$$



(a) Backbone (28 dims).



(b) DiT (78 dims).

Figure 5: CCA canonical-correlation spectra (PCA truncation + whitening.

Canonical correlations. CCA is obtained via SVD:

$$\hat{X}^\top \hat{Y} = U \operatorname{diag}(\rho_1, \ldots, \rho_k)\, V^\top, \tag{11}$$

where $k = \min(k_x, k_y)$ and $\rho_j \in [0, 1]$ are canonical correlations.

A.5 CCA Canonical-correlation Spectra and Original-space Aligned Energy

We visualize the PCA-whitened CCA canonical-correlation spectra and report how much *original-space* feature energy lies in highly aligned CCA directions. Importantly, the "CCA-aligned subspace" here is a post-hoc linear construct and should not be conflated with the learned *shared/unique* factors of the SAE.

Canonical-correlation spectra. Figure 5 shows the canonical correlations $\{\rho_j\}$ for backbone-level features (28 PCA-whitened dimensions) and DiT-level features (78 PCA-whitened dimensions). DiT yields a much larger set of near-perfectly aligned directions, consistent with increased decision-level isomorphism.

Original-space aligned-subspace energy (thresholded by $\rho > \tau$). Let $X \in \mathbb{R}^{n \times d_x}$ and $Y \in \mathbb{R}^{n \times d_y}$ be centered features. After PCA truncation and whitening (Appendix A.4), we obtain whitened features $\tilde{X} = X U_X \Lambda_X^{-\frac{1}{2}}$ and $\tilde{Y} = Y U_Y \Lambda_Y^{-\frac{1}{2}}$. Running CCA on $(\tilde{X}, \tilde{Y})$ yields canonical directions A, B and canonical correlations $\rho_1 \geq \cdots \geq \rho_k$.

For a threshold τ (we use $\tau = 0.8$), define the index set

$$\mathcal{I}_\tau = \{j : \rho_j > \tau\}.$$

Map the selected canonical directions back to the original feature coordinates:

$$Q_X = U_X \Lambda_X^{-\frac{1}{2}} A_{\mathcal{I}_\tau}, \qquad Q_Y = U_Y \Lambda_Y^{-\frac{1}{2}} B_{\mathcal{I}_\tau}.$$

Let $\mathrm{orth}(\cdot)$ return an orthonormal basis for the column span (e.g., via QR), and define $\bar{Q}_X = \mathrm{orth}(Q_X)$ and $\bar{Q}_Y = \mathrm{orth}(Q_Y)$. We then measure the fraction of *original-space* energy captured by the CCA-aligned subspace as

$$E_X^{\mathrm{full}} = \frac{1}{n}\|X\|_F^2, \quad E_X^{\mathrm{align}}(\tau) = \frac{1}{n}\|X\bar{Q}_X\|_F^2, \quad \mathrm{Frac}_X^{\mathrm{align}}(\tau) = \frac{E_X^{\mathrm{align}}(\tau)}{E_X^{\mathrm{full}}},$$

and analogously for Y using $\bar{Q}_Y$.

Numerical summary (used in Table 2). Backbone level (28 dims): top-10 ρ are 0.94, 0.91, 0.87, 0.85, 0.82, 0.77, 0.73, 0.71, 0.71, 0.69 (mean@10 $= 0.800$). With $\tau = 0.8$, the aligned-energy fractions are 28.6% (VLM) and 55.6% (ViT).

DiT level (78 dims): top-10 ρ are 0.996, 0.994, 0.986, 0.983, 0.980, 0.975, 0.968, 0.954, 0.946, 0.941 (mean@10 ≈ 0.972). With $\tau = 0.8$, the aligned-energy fractions are 53.4% (VLM) and 77.1% (ViT).

A.6 Shared–Unique SAE: Full Objective and Regularizers

This section specifies the Shared–Unique SAE used in RQ1. SAE is trained in standardized feature space (Appendix A.2).

A.6.1 Encoders and additive linear decoders

For a minibatch $\{(x^{(i)}, y^{(i)})\}_{i=1}^B$,

$$z_s^x = f_s^x(x) \in \mathbb{R}^{B\times d_s}, \qquad z_u^x = f_u^x(x) \in \mathbb{R}^{B\times d_u}, \tag{12}$$

$$z_s^y = f_s^y(y) \in \mathbb{R}^{B\times d_s}, \qquad z_u^y = f_u^y(y) \in \mathbb{R}^{B\times d_u}. \tag{13}$$

We use MLP encoders (ReLU) and additive linear decoders

$$\hat{x} = W_s^x z_s^x + W_u^x z_u^x + \mathbf{1}(b^x)^\top, \tag{14}$$

$$\hat{y} = W_s^y z_s^y + W_u^y z_u^y + \mathbf{1}(b^y)^\top, \tag{15}$$

where $W_s^x \in \mathbb{R}^{d\times d_s}$, $W_u^x \in \mathbb{R}^{d\times d_u}$ (and similarly for y), and $b^x, b^y \in \mathbb{R}^d$.

Shared-only reconstructions (self).

$$\hat{x}_s = W_s^x z_s^x + \mathbf{1}(b^x)^\top, \qquad \hat{y}_s = W_s^y z_s^y + \mathbf{1}(b^y)^\top. \tag{16}$$

Shared-only reconstructions (cross).

$$\hat{x}_{s\leftarrow y} = W_s^x z_s^y + \mathbf{1}(b^x)^\top, \qquad \hat{y}_{s\leftarrow x} = W_s^y z_s^x + \mathbf{1}(b^y)^\top. \tag{17}$$

Mixed reconstructions (cross-shared + self-unique).

$$\hat{x}_{\mathrm{mix}} = W_s^x z_s^y + W_u^x z_u^x + \mathbf{1}(b^x)^\top, \tag{18}$$

$$\hat{y}_{\mathrm{mix}} = W_s^y z_s^x + W_u^y z_u^y + \mathbf{1}(b^y)^\top. \tag{19}$$

A.6.2 Loss terms

The total objective is

$$\mathcal{L} = \lambda_{\mathrm{rec}}\mathcal{L}_{\mathrm{rec}} + \lambda_{\mathrm{sh}}\mathcal{L}_{\mathrm{sh}} + \lambda_{\mathrm{cross}}\mathcal{L}_{\mathrm{cross}} + \lambda_{\mathrm{vic}}\mathcal{L}_{\mathrm{vic}} + \lambda_{\mathrm{ort}}\mathcal{L}_{\mathrm{ort}} + \lambda_{\mathrm{sp}}\mathcal{L}_{\mathrm{sp}}. \tag{20}$$

(1) Full reconstruction.

$$\mathcal{L}_{\mathrm{rec}} = \frac{1}{Bd}\|\hat{x} - x\|_F^2 + \frac{1}{Bd}\|\hat{y} - y\|_F^2. \tag{21}$$

(2) Shared-only reconstruction (self).

$$\mathcal{L}_{\mathrm{sh}} = \frac{1}{Bd}\|\hat{x}_s - x\|_F^2 + \frac{1}{Bd}\|\hat{y}_s - y\|_F^2. \tag{22}$$

(3) Cross shared-only reconstruction (interchangeability).

$$\mathcal{L}_{\text{cross}} = \frac{1}{Bd}\|\hat{x}_{s\leftarrow y} - x\|_F^2 + \frac{1}{Bd}\|\hat{y}_{s\leftarrow x} - y\|_F^2. \tag{23}$$

A.6.3 VICReg-style anti-collapse on shared latents

We apply VICReg-style constraints (Bardes et al., 2021) on (z_s^x, z_s^y):

$$\mathcal{L}_{\text{vic}} = \alpha\,\mathcal{L}_{\text{inv}} + \beta\,(\mathcal{L}_{\text{var}}(z_s^x) + \mathcal{L}_{\text{var}}(z_s^y)) + \gamma\,(\mathcal{L}_{\text{cov}}(z_s^x) + \mathcal{L}_{\text{cov}}(z_s^y))\,. \tag{24}$$

Invariance. Let $\mathrm{BN}(\cdot)$ standardize each dimension within the minibatch (zero mean, unit std). We use

$$\mathcal{L}_{\text{inv}} = \frac{1}{B}\,\|\mathrm{BN}(z_s^x) - \mathrm{BN}(z_s^y)\|_F^2\,. \tag{25}$$

Variance (hinge). Let $\mathrm{Std}(Z) \in \mathbb{R}^{d_s}$ denote per-dimension standard deviation across the batch. With margin $\nu > 0$,

$$\mathcal{L}_{\text{var}}(Z) = \frac{1}{d_s}\sum_{j=1}^{d_s} \max\bigl(0, \nu - \mathrm{Std}(Z)_j\bigr)^2. \tag{26}$$

Covariance (decorrelation). Let $Z_c = Z - \frac{1}{B}\mathbf{1}\mathbf{1}^\top Z$ be batch-centered and

$$\mathrm{Cov}(Z) = \frac{1}{B-1}Z_c^\top Z_c. \tag{27}$$

Then

$$\mathcal{L}_{\text{cov}}(Z) = \frac{1}{d_s}\,\bigl\|\mathrm{OffDiag}\bigl(\mathrm{Cov}(Z)\bigr)\bigr\|_F^2\,, \tag{28}$$

where $\mathrm{OffDiag}(\cdot)$ zeros the diagonal entries.

A.6.4 Shared–unique separability (orthogonality)

We penalize cross-covariance between shared and unique latents within each branch. For batch-centered A_c, B_c,

$$\mathrm{Cov}(A, B) = \frac{1}{B-1}A_c^\top B_c. \tag{29}$$

Then

$$\mathcal{L}_{\text{ort}} = \|\mathrm{Cov}(z_s^x, z_u^x)\|_F^2 + \|\mathrm{Cov}(z_s^y, z_u^y)\|_F^2\,. \tag{30}$$

A.6.5 Sparsity on unique latents

We encourage compact residual coding via ℓ_1 sparsity:

$$\mathcal{L}_{\text{sp}} = \frac{1}{B}\|z_u^x\|_1 + \frac{1}{B}\|z_u^y\|_1. \tag{31}$$

A.7 SAE Metrics

All R^2 scores are computed in standardized feature space. For any reconstruction $\hat{x}$ of x:

$$R^2(\hat{x}; x) = 1 - \frac{\mathrm{MSE}(\hat{x}, x)}{\mathrm{Var}(x)}, \tag{32}$$

$$\mathrm{MSE}(\hat{x}, x) = \frac{1}{Bd}\|\hat{x} - x\|_F^2, \qquad \mathrm{Var}(x) = \frac{1}{Bd}\|x - \bar{x}\|_F^2, \tag{33}$$

where $\bar{x}$ is the per-dimension mean computed consistently with the standardization protocol (Appendix A.2).

We report:

$$R^2_{\text{full}}(x) = R^2(\hat{x}; x), \qquad R^2_{\text{sh}}(x) = R^2(\hat{x}_s; x), \qquad R^2_{\text{cross}}(x) = R^2(\hat{x}_{s\leftarrow y}; x), \tag{34}$$

(and analogously for y), and define the self–cross gap

$$\Delta_{\text{cross}}(x) = R^2_{\text{sh}}(x) - R^2_{\text{cross}}(x), \qquad \Delta_{\text{cross}}(y) = R^2_{\text{sh}}(y) - R^2_{\text{cross}}(y). \tag{35}$$

Table 9: SAE sweep results (standardized space metrics). We report reconstruction quality (full/shared-only), shared-space alignment ($\mathrm{CKA}_{\mathrm{shared}}$), and interchangeability via cross reconstruction and the self–cross gap.

Feature	use_raw_mse	cross_weight	$R^2_{\mathrm{full}}(x)$	$R^2_{\mathrm{full}}(y)$	$R^2_{\mathrm{shared}}(x)$	$R^2_{\mathrm{shared}}(y)$	$\mathrm{CKA}_{\mathrm{shared}}$	$R^2_{\mathrm{cross}}(x{\leftarrow}z^y_s)$ / $R^2_{\mathrm{cross}}(y{\leftarrow}z^x_s)$	$\Delta_{\mathrm{cross}}(x)$ / $\Delta_{\mathrm{cross}}(y)$
Backbone features ($\mathrm{CKA}_{\mathrm{orig}}$=0.2125)									
backbone	False	0.0	0.787	0.901	0.641	0.784	0.982	0.492 / 0.559	0.149 / 0.226
backbone	False	0.1	0.786	0.902	0.634	0.784	0.981	0.537 / 0.623	0.098 / 0.160
backbone	False	0.2	0.784	0.900	0.629	0.768	0.982	0.541 / 0.634	0.087 / 0.134
backbone	False	0.5	0.779	0.897	0.626	0.768	0.981	0.582 / 0.687	0.045 / 0.081
backbone	False	1.0	0.780	0.896	0.623	0.772	0.981	0.598 / 0.715	**0.025 / 0.057**
backbone	True	0.0	0.785	0.900	0.633	0.781	0.984	0.522 / 0.602	0.111 / 0.180
backbone	True	0.1	0.783	0.901	0.623	0.774	0.983	0.541 / 0.627	0.082 / 0.146
backbone	True	0.2	0.781	0.900	0.625	0.772	0.984	0.554 / 0.648	0.072 / 0.124
backbone	True	0.5	0.779	0.892	0.618	0.763	0.981	0.579 / 0.685	0.039 / 0.078
backbone	True	1.0	0.779	0.894	0.622	0.761	0.982	0.595 / 0.710	**0.026 / 0.051**
DiT features ($\mathrm{CKA}_{\mathrm{orig}}$=0.5369)									
dit	False	0.0	0.801	0.926	0.617	0.825	0.988	0.534 / 0.754	0.071 / 0.083
dit	False	0.1	0.799	0.922	0.617	0.826	0.986	0.546 / 0.763	0.063 / 0.071
dit	False	0.2	0.800	0.924	0.614	0.822	0.986	0.552 / 0.774	0.048 / 0.061
dit	False	0.5	0.800	0.922	0.613	0.822	0.986	0.566 / 0.791	0.031 / 0.047
dit	False	1.0	0.802	0.922	0.622	0.831	0.984	0.580 / 0.811	**0.021 / 0.042**
dit	True	0.0	0.800	0.923	0.612	0.816	0.987	0.537 / 0.754	0.062 / 0.075
dit	True	0.1	0.801	0.922	0.614	0.816	0.988	0.548 / 0.766	0.050 / 0.066
dit	True	0.2	0.796	0.921	0.606	0.814	0.986	0.549 / 0.770	0.044 / 0.057
dit	True	0.5	0.798	0.922	0.609	0.821	0.987	0.565 / 0.790	0.032 / 0.044
dit	True	1.0	0.797	0.920	0.609	0.823	0.985	0.576 / 0.805	**0.018 / 0.033**

A.8 Output-Space Variance Attribution

With additive decoder contributions

$$x_s = W_s^x z_s^x, \qquad x_u = W_u^x z_u^x, \qquad \varepsilon_x = x - (x_s + x_u + \mathbf{1}(b^x)^\top), \tag{36}$$

we decompose (in standardized space)

$$\mathrm{Var}(x) = \mathrm{Var}(x_s) + \mathrm{Var}(x_u) + 2\,\mathrm{Cov}(x_s, x_u) + \mathrm{Var}(\varepsilon_x), \tag{37}$$

(and similarly for y). We report all four components; shared/unique percentages do not necessarily sum to 100% when covariance/residual terms are non-zero.

A.9 Shuffled-Pair Control for Shared-Space Saturation

Shared-space similarity (e.g., cosine similarity or CKA computed on the SAE shared representations) often saturates by design because the SAE objective explicitly enforces invariance between paired shared latents. To verify that this saturation is not due to a trivial solution, we perform a *shuffled-pair control*: we randomly permute pairings (x_i, y_i) while keeping the marginals of x and y unchanged, retrain SAE with the same hyperparameters, and re-compute shared-space and original-space similarity.

We expect two qualitative outcomes: (i) shared-space similarity should decrease under shuffled pairing, and (ii) original-space CKA should collapse toward zero, confirming that high shared-space alignment relies on correct pairings and is not a trivial artifact.

A.10 SAE Hyperparameter Sweep

We sweep SAE settings over `use_raw_mse`$\in$ {False, True} and `cross_weight`$\in \{0.0, 0.1, 0.2, 0.5, 1.0\}$, and report original-space alignment and interchangeability metrics in standardized space.

A.11 Rule-based Representation-only Gating from SAE Energy Decomposition

We build handcrafted gating rules from the Shared–Unique SAE decomposition (Appendix A.6). For each branch, let the additive decoder contributions (in standardized feature space) be

$$x_s = W_s^x z_s^x, \quad x_u = W_u^x z_u^x, \qquad y_s = W_s^y z_s^y, \quad y_u = W_u^y z_u^y.$$

We define squared-ℓ_2 "energy" in shared/unique subspaces as

$$E^s_{\mathrm{vlm}} = \|x_s\|_2^2, \quad E^u_{\mathrm{vlm}} = \|x_u\|_2^2, \qquad E^s_{\mathrm{vit}} = \|y_s\|_2^2, \quad E^u_{\mathrm{vit}} = \|y_u\|_2^2,$$

and $E^{\mathrm{total}} = E^s + E^u$. We use a small constant $\epsilon > 0$ for numerical stability.

Table 10: Threshold sweep for rule-based gating. Fill with the same evaluation metrics used in Table 5.

τ	navtest PDMS(%)
0.5	89.73
0.6	89.75
0.7	89.92
0.8	89.90
0.9	89.89

Indicators. We derive four indicators (symmetrically for both branches):

$$r^u = \frac{E^u}{E^{\text{total}} + \epsilon}, \quad r^s = \frac{E^s}{E^{\text{total}} + \epsilon}, \quad u = \frac{E^u}{E^s + \epsilon}, \quad d = \frac{E^s}{E^s + E^u + \epsilon}, \quad \bar{d} = \frac{d_{\text{vlm}} + d_{\text{vit}}}{2}.$$

Decision convention. Each strategy produces a signed score s (positive favors VLM; negative favors ViT), and outputs the decision

$$\text{choose VLM if } s > 0, \quad \text{otherwise choose ViT.}$$

(i) More-unique wins. We compare uniqueness strength:

$$s_1 = u_{\text{vlm}} - u_{\text{vit}}.$$

(ii) Shared-dominant conditional (hard regime). Given a shared-dominance threshold τ,

$$s_2(\tau) = \begin{cases} r^s_{\text{vlm}} - r^s_{\text{vit}}, & \bar{d} > \tau, \\ u_{\text{vlm}} - u_{\text{vit}}, & \text{otherwise.} \end{cases}$$

(iii) Smoothed shared-dominance (sigmoid regime). To reduce sensitivity near the threshold, we replace the hard indicator by a sigmoid weight

$$w(\bar{d}; \tau) = \sigma\big(\kappa(\bar{d} - \tau)\big) = \frac{1}{1 + \exp\big(-\kappa(\bar{d} - \tau)\big)} \in (0, 1),$$

where we fix $\kappa = 5$ (corresponding to `SOFT_LABEL_SCALE=5`). We then define

$$s_3(\tau) = w(\bar{d}; \tau)\,(r^s_{\text{vlm}} - r^s_{\text{vit}}) + \big(1 - w(\bar{d}; \tau)\big)\,(u_{\text{vlm}} - u_{\text{vit}}).$$

(iv) ViT-prior fallback. We default to ViT and switch to VLM only when the scenario is strongly shared-dominant:

$$\text{choose VLM if } \bar{d} > \tau_{\text{strong}} \text{ and } s_3(\tau) > 0; \quad \text{else choose ViT.}$$

This strategy is intentionally conservative to avoid over-switching. (*To reproduce this variant,* τ_{strong} *must be specified.*)

Threshold sweep and main setting. We sweep the shared-dominance threshold over

$$\tau \in \{0.5,\, 0.6,\, 0.7,\, 0.8,\, 0.9\},$$

and select the best-performing value under the same evaluation protocol used for Table 5. The main text reports results for $\tau = 0.7$, which achieves the highest score among the tested thresholds. For the smoothed variant, we use the same sweep with fixed $\kappa = 5$.

A.12 Rule-based Gating: Threshold Sweep

We report the performance of the shared-dominance threshold sweep for the rule-based gates. The main text uses $\tau = 0.7$.

A.13 Learned Representation-only Gating Models and t-SNE Diagnostics

We formulate gating as supervised learning. For each scenario, we construct inputs from both branches' representations at either the backbone level or the DiT level. Common feature constructions include concatenation $[x; y]$, difference $(x - y)$, and their combination $[x; y; x - y]$.

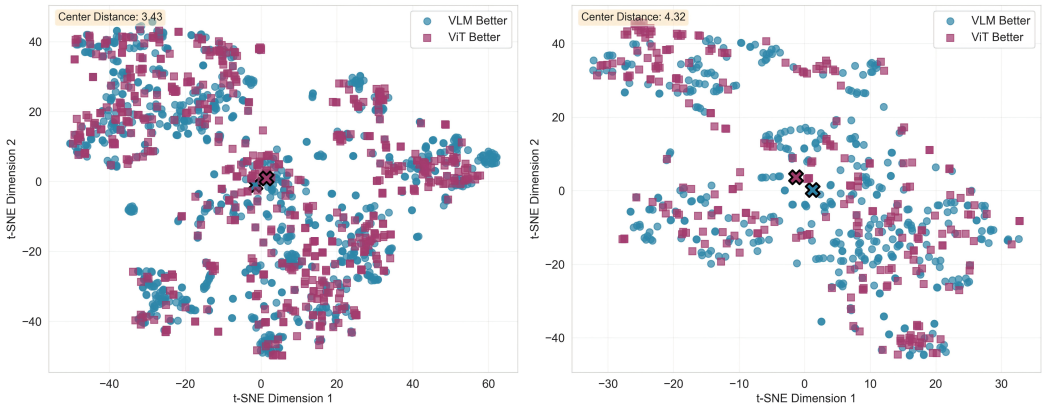


Figure 6: t-SNE of backbone- and DiT-level features colored by whether VLM outperforms ViT for the scenario. The classes are not separable, suggesting intrinsic difficulty for representation-only gating. (Left: backbone-level features; Right: DiT-level features.)

Labels. The binary label indicates which branch yields better closed-loop performance for that scenario (VLM-better vs. ViT-better), computed from the evaluation score used in Table 5. Ties can be discarded or broken deterministically.

Model families. We evaluate:

- **Tree-based models:** Random Forest; Gradient Boosting / GBDT variants.
- **MLP gate:** multi-layer fully-connected network with BatchNorm and Dropout, trained with binary cross-entropy, sigmoid output.
- **Token-aware attention gate:** self-attention encoder over the VLM token sequence; cross-attention using the vision-only global feature as a query over VLM keys/values; MLP head for classification. This explicitly uses the long VLM token sequence rather than pooled features alone.

t-SNE diagnostic. To qualitatively assess separability, we apply t-SNE on backbone-level and DiT-level representations and color points by the binary label (VLM-better vs. ViT-better). Poor class separation in both spaces (Fig. 6) is consistent with the difficulty of representation-only gating, though t-SNE is used only as a visualization tool rather than a definitive test.

B Offline Trajectory-quality Score in NAVSIM (PDMS v1 and EPDMS v2)

We use the planning-oriented NAVSIM benchmark and adopt the official Predictive Driver Model Score (PDMS) from NAVSIM v1 as our primary offline trajectory-quality score $s(\cdot,\cdot)$; higher is better. PDMS is a pseudo closed-loop metric that holistically assesses safety, comfort, and progress via multiplicative penalties and a weighted average:

$$\mathrm{PDMS} \;=\; \mathrm{NC} \times \mathrm{DAC} \times \left(\frac{5 \cdot \mathrm{EP} + 5 \cdot \mathrm{TTC} + 2 \cdot \mathrm{C}}{12} \right), \tag{38}$$

where NC denotes no at-fault collisions, DAC drivable-area compliance, EP ego progress, TTC time-to-collision within bound, and C comfort.

NAVSIM v2: Extended PDMS (EPDMS). NAVSIM v2 extends PDMS to improve coverage and fairness of open-loop planning evaluation. Compared to NAVSIM v1, EPDMS introduces additional weighted subscores (lane keeping and extended comfort variants), additional multiplier penalties (driving direction compliance and traffic light compliance), and a false-positive penalty filtering scheme.

False-positive penalty filtering. To reduce false-positive penalties, NAVSIM v2 disables a penalty when the human agent is also responsible for the corresponding violation. Formally, for a metric m,

Table 11: EPDMS composition in NAVSIM v2 (new metrics relative to v1 are highlighted).

Metric	Weight	Range
No at-fault Collisions (NC)	multiplier	$\{0, \frac{1}{2}, 1\}$
Drivable Area Compliance (DAC)	multiplier	$\{0, 1\}$
Driving Direction Compliance (DDC)	multiplier	$\{0, \frac{1}{2}, 1\}$
Traffic Light Compliance (TLC)	multiplier	$\{0, 1\}$
Ego Progress (EP)	5	$[0, 1]$
Time to Collision (TTC) within bound	5	$\{0, 1\}$
Lane Keeping (LK)	2	$\{0, 1\}$
History Comfort (HC)	2	$\{0, 1\}$
Extended Comfort (EC)	2	$\{0, 1\}$

define

$$\text{filter}_m(\text{agent}, \text{human}) = \begin{cases} 1.0, & \text{if } m(\text{human}) = 0, \\ m(\text{agent}), & \text{otherwise.} \end{cases} \tag{39}$$

Intuitively, if the human baseline also triggers the violation, the metric is neutralized (set to 1.0) rather than penalizing the planner.

EPDMS definition. With the above filtering, EPDMS is defined as

$$\text{EPDMS} = \left(\prod_{m \in \{\text{NC,DAC,DDC,TLC}\}} \text{filter}_m(\text{agent}, \text{human}) \right) \tag{40}$$

$$\cdot \left(\frac{\sum_{m \in \{\text{TTC,EP,HC,LK,EC}\}} w_m \cdot \text{filter}_m(\text{agent}, \text{human})}{\sum_{m \in \{\text{TTC,EP,HC,LK,EC}\}} w_m} \right), \tag{41}$$

where w_m are the weights listed in Table 11.

Pseudo closed-loop aggregation in NAVSIM v2. NAVSIM v1 computes metrics after a 4-second non-reactive simulation rollout (background actors follow recorded futures; ego follows the planned trajectory via a controller). NAVSIM v2 uses a two-stage aggregation to better approximate closed-loop behavior while remaining open-loop: (i) a first-stage score is computed on an initial 4-second scene; (ii) multiple follow-up scenes (precomputed rollouts starting from the same initial scene but with different end states) are also scored, and then aggregated with weights given by a Gaussian kernel based on how close each follow-up scene's start state is to the submitted planner's first-stage end state. Finally, the first-stage score and the aggregated second-stage score are multiplied to obtain the final aggregated EPDMS score.

C RQ2 Win Counting Protocol and Thresholds

For scenario-level complementarity, we compare policies via the per-scenario advantage

$$\Delta_{r,i} = s(\text{VLM}, r, i) - s(\text{ViT}, r, i),$$

where $s(m, r, i)$ is the NAVSIM v1 PDMS score of policy m on scenario i under random seed $r \in \{1, 2, 3\}$ (Appendix B). A per-seed significant win is defined as $|\Delta_{r,i}| > \tau$.

D Additional Cross-model Evidence Beyond InternVL3-2B

To test whether the shared-plus-unique structure is specific to a single VLM, we additionally trained InternVL3-8B and Qwen3VL-8B on NAVSIM with the same three-stage recipe as the main model and repeated the RQ1 analysis. We also repeated the same style of analysis on AsyncDrive/nuPlan, using GameFormer as the fast branch and Llama2-13B as the slow branch. For the NAVSIM runs, the three CKA values are reported in the order of VLM paired with ViT, ResNet, and EVA-CLIP. Specifically, for InternVL3-8B, backbone CKA is $0.26/0.23/0.24$ and DiT CKA is $0.49/0.48/0.48$; for Qwen3VL-8B, backbone CKA is $0.30/0.29/0.31$ and DiT CKA is $0.55/0.54/0.54$. On AsyncDrive/nuPlan, the rebuttal analysis reported a single planner-level CKA summary of 0.66.

Table 12: Additional SAE/CCA evidence for RQ1 across larger VLMs and an external planner stack. Higher R^2_{cross} and CCA indicate stronger shared structure. For AsyncDrive/nuPlan, the rebuttal analysis reported only a single planner-level summary rather than separate backbone and DiT values.

Setting	PDMS	R^2_{cross} (backbone)	R^2_{cross} (DiT / planner)	CCA mean@10	Note
InternVL3-8B on NAVSIM	90.4	0.49 / 0.62	0.54 / 0.78	0.81 / 0.97	CKA in text
Qwen3VL-8B on NAVSIM	90.7	0.44 / 0.60	0.52 / 0.81	0.84 / 0.95	CKA in text
AsyncDrive / nuPlan	-	-	0.61 / 0.73	- / 0.88	planner-level summary only

Table 13: **Domain-adaptation control on NAVSIM.** *Main* denotes VLM-full-DA vs. ViT-large. *NAVSIM-DA* denotes VLM-NAVSIM-DA vs. ViT-NAVSIM-DA.

Setting	VLM PDMS ↑	ViT PDMS ↑	Oracle ↑	Complex wins
Main	90.80	88.88	93.58	139 / 187
NAVSIM-DA	90.64	87.45	93.20	133 / 187

E Token-level robustness to mean pooling.

Because the main paper uses pooled VLM features for tractable full-dataset statistics, we additionally ran token-level / non-global-pooling analyses over 20 random trials on the largest subset that fits our hardware (roughly one-tenth of the full dataset). CKA and SAE statistics remain within about 5% of the main values, while CCA mean@10 fluctuates by roughly 5–10% in some runs. Crucially, the qualitative conclusion is unchanged: VLM and vision-only backbones differ more strongly at the backbone level, become more aligned after policy learning, and still preserve non-shared residual factors.

F Domain-Adaptation Control

Motivation. The main experiments compare practical VLA-style and vision-only E2E representation pipelines under the same downstream planner. However, the upstream histories are not identical: the VLM branch uses a driving-domain adaptation stage, whereas the vision-only branch starts from a strong public visual checkpoint and is trained through the downstream planner. We therefore add a data-restricted control to test whether the observed VLM–ViT complementarity is mainly caused by this domain-adaptation asymmetry.

Controlled setups. We evaluate two additional variants. First, **VLM-NAVSIM-DA** restricts the VLM driving adaptation stage to NAVSIM training data and its available text/trajectory annotations, instead of using the full multi-dataset RecogDrive-style corpus. The adapted VLM encoder is then frozen and inserted into the same downstream planner training pipeline as the original VLM branch. Second, **ViT-NAVSIM-DA** initializes from the same public ViT-large checkpoint as the original vision-only baseline, performs MAE adaptation on NAVSIM training images only, discards the MAE decoder, and then follows the same downstream planner training protocol as the original ViT-large branch. All variants use the same NAVSIM split, trajectory decoding format, downstream planner/action architecture, and evaluation protocol. No NAVSIM test images or evaluation labels are used in the adaptation stage.

Why ViT is not trained from scratch. We initialize ViT-NAVSIM-DA from the public ViT-large checkpoint rather than training it from scratch. The purpose of this control is to test the effect of *in-domain adaptation*, not to remove generic large-scale visual pretraining. Training ViT from scratch on NAVSIM would introduce a much stronger and less realistic pretraining-budget asymmetry, since the VLM branch necessarily starts from a large pretrained multimodal checkpoint.

Results. Table 13 summarizes the control. Restricting the VLM adaptation to NAVSIM causes only a small performance change compared with VLM-full-DA. MAE adaptation also gives the vision-only branch a comparable in-domain adaptation stage. More importantly, the VLM–ViT complementarity remains: the VLM branch still has a clear advantage in complex semantic and interaction-heavy scenes, while the cross-model best-of-two score remains substantially above either single branch. This suggests that the observed complementarity is not explained away by additional driving-domain exposure alone.

Interpretation and scope. This control should not be interpreted as a fully symmetric pretraining comparison. VLM and ViT differ in architecture, modality, and upstream objectives, and a perfectly matched adaptation budget is not well-defined. Instead, the control addresses a concrete confound: whether the complementarity observed in the main experiments disappears once the vision-only branch also receives in-domain adaptation and the VLM adaptation is restricted to the same dataset. The persistence of the complex-scene advantage and the cross-model oracle gain indicates that the main conclusion is not an artifact of extra driving-domain data alone.

G Scene Taxonomy for Major-Decision-Mismatch Cases

Motivation. RQ2 analyzes 279 major-decision-mismatch cases, where one branch makes a substantially worse decision while the other remains broadly reasonable under a strict threshold $\tau = 0.9$. Because these cases are heterogeneous, we use a coarse scene taxonomy to understand where VLM–vision complementarity arises. The taxonomy is intended for aggregate behavioral analysis rather than fine-grained semantic benchmarking.

Scene-description generation. For each of the 279 cases, we first generate a driving-scene description following the VLM annotation format used in RecogDrive-style NAVSIM training data. Given the front-view image, ego history, navigation command, and available scene metadata, the annotation prompts ask for planning-relevant factors such as surrounding vehicles and pedestrians, road markings, traffic lights, congestion, vulnerable road users, influential objects, and the likely short-horizon driving behavior. An example annotation includes scene-level descriptions, congestion status, road-marking interpretation, influential-object analysis, vulnerable-road-user detection, and a short speed/path decision explanation. These descriptions are used only for post-hoc scenario grouping and are not used to train the policies or the trajectory scorer.

Coarse grouping protocol. We then use GPT-5.4 to assign each case to one primary scene type based on the generated description and the driving context. When multiple factors are present, we assign the type that is most relevant to the trajectory decision difference between the two branches. The authors manually checked the AI-assisted annotations and found no obvious errors or omissions. The six coarse categories are:

Table 14: **Coarse scene taxonomy for the 279 major-decision-mismatch cases.** The taxonomy is used only for RQ2 behavioral analysis.

Category	Definition
Simple lane keeping	Straightforward lane-following cases with no salient semantic ambiguity, unusual object, or interaction-heavy event.
Intersection semantics	Cases where traffic lights, intersections, crossing structure, right-of-way, or turn/straight semantics strongly influence the decision.
Curve / merge / ramp	Cases involving curved roads, ramp-like geometry, lane merging, or route-following ambiguity.
Work zone / cones	Cases where cones, construction areas, blocked lanes, or temporary traffic layouts affect the feasible path.
Narrow / unclear boundary	Cases with narrow drivable space, weak or ambiguous lane boundaries, parked vehicles near the ego path, or unclear road edges.
Occlusion / dense clutter	Cases with occluded agents, dense traffic, pedestrians, cluttered urban layouts, or limited visibility of relevant interactions.

Simple vs. complex split. We additionally merge the six categories into a binary simple/complex split. A case is labeled *simple* only when the required trajectory decision is straight lane following and the scene description contains no obvious out-of-distribution object, temporary event, semantic ambiguity, dense interaction, or unclear drivable boundary. All other cases are labeled *complex*. This definition is intentionally conservative: rather than trying to precisely quantify all forms of complexity, we only mark a scene as simple when it is clearly simple. Under this protocol, the simple subset contains 92 cases and the complex subset contains 187 cases.

Use in RQ2. This taxonomy yields the scene-level breakdown reported in Sec. 5.1. In the 279 major-decision-mismatch cases, VLM wins 176 cases and ViT wins 103. ViT wins more often only

Table 15: Scenario taxonomy for the 279 major-difference cases ($\tau = 0.9$). Percentages are relative to the 279 categorized cases.

Scenario type	Count	Share	VLM better	ViT better	VLM win rate	Complexity
simple_lane_keep	92	33.0%	37	55	40.2%	Simple
intersection_semantic	78	28.0%	58	20	74.4%	Complex
curve_merge_ramp	38	13.6%	23	15	60.5%	Complex
workzone_cone	15	5.4%	12	3	80.0%	Complex
narrow_unclear_boundary	44	15.8%	34	10	77.3%	Complex
occlusion_dense_clutter	12	4.3%	12	0	100.0%	Complex
Simple (merged)	92	33.0%	37	55	40.2%	-
Complex (merged)	187	67.0%	139	48	74.3%	-

Table 16: Behavioral and safety diagnostics for VLM vs. ViT on NAVSIM. The significant-difference subset contains scenarios where the two branches differ substantially in score.

Metric	VLM (full set)	ViT (full set)	VLM (significant subset)	ViT (significant subset)
Mean path speed (m/s)	5.31($\pm$ 0.05)	5.17($\pm$ 0.02)	5.74($\pm$ 0.11)	5.43($\pm$ 0.04)
Path length (m)	18.60($\pm$ 0.07)	18.09($\pm$ 0.05)	20.08($\pm$ 0.14)	19.02($\pm$ 0.11)
Min longitudinal acceleration (m/s^2)	-2.87($\pm$ 0.01)	-2.69($\pm$ 0.01)	-2.99($\pm$ 0.03)	-2.53($\pm$ 0.02)
Cumulative deceleration	4.94($\pm$ 0.02)	4.63($\pm$ 0.02)	5.26($\pm$ 0.02)	4.38($\pm$ 0.02)
Hard-brake segments	0.49($\pm$ 0.01)	0.44($\pm$ 0.00)	0.49($\pm$ 0.01)	0.36($\pm$ 0.00)
No-collision score (NC)	97.9($\pm$ 0.01)	97.5($\pm$ 0.01)	-	-
Drivable-area compliance (DAC)	97.3($\pm$ 0.01)	97.1($\pm$ 0.01)	-	-
Time-to-collision (TTC)	94.9($\pm$ 0.01)	93.6($\pm$ 0.01)	-	-
Minimum obstacle distance (m)	2.12($\pm$ 0.02)	2.11($\pm$ 0.02)	2.7($\pm$ 0.02)	1.8($\pm$ 0.01)

in simple lane keeping, whereas VLM dominates the semantically complex and interaction-heavy categories. The taxonomy should therefore be interpreted as a coarse diagnostic of where the two branches differ, not as a claim that the six categories exhaust all possible driving factors.

H Scenario Taxonomy for Major-difference Cases

For the semantic scenario-breakdown analysis in RQ2, we use a stricter threshold $\tau = 0.9$ than the stability-oriented win-counting thresholds in the main text. This subset is intended to isolate cases where one branch makes a major decision error while the other remains broadly reasonable. Under this criterion, we obtain 279 scenarios, of which VLM wins 176 and ViT wins 103.

The long-tail categories most favorable to the VLM branch are intersections / semantic decisions ($58/78$), work zones / cones ($12/15$), unclear boundaries / narrow roads ($34/44$), and occlusion / dense clutter ($12/12$). By contrast, ViT wins more often in the simple lane-keeping subset ($55/92$), consistent with the main-text claim that VLM-specific residual factors are particularly useful in scenarios that require semantic interpretation, ambiguity resolution, or stronger multi-agent interaction handling.

I Behavioral Statistics and Safety Diagnostics

Table 16 provides the full behavioral and safety diagnostics summarized in RQ2, with standard deviations computed over five random seeds. The VLM branch is not only faster on average, but also exhibits a distinct progress–braking trade-off: it advances further while showing stronger braking-side responses when required.

Intervention-style evidence from SAE-decomposed latents. We also tested whether the residual factors isolated by the Shared–Unique SAE are behaviorally useful by training policies on different latent combinations. On NAVSIM, shared-only features reach 83.0 PDMS; shared + ViT-unique reaches 86.1; shared + VLM-unique reaches 86.9; and shared + both unique reaches 87.5. The effect is even clearer in the 187 semantically complex / interaction-heavy cases: shared + VLM-unique yields high-score outcomes in 86 scenes, compared with 37 for shared + ViT-unique and 33 for shared-only. This supports the interpretation that VLM-specific residual factors are especially useful in complex scenarios rather than being purely representational artifacts.

Table 17: Tail-risk analysis of severe switcher failures. Exact VLM invocation for the more conservative setting was not reported in the rebuttal, so it is marked with "-".

Setting	Severe failures ($> 20\%$ gap)	Severe failures ($> 50\%$ gap)	VLM invocation
DualDriveVLA (default)	50	30	about 15%
More conservative fallback	40	21	-

Tail-risk of the scorer-based switcher. We do not interpret the scorer as a certified safety module. Instead, we use it as a practical selector whose main operating knob is the fallback threshold: raising the threshold invokes the slow VLM branch more often, improving worst-case behavior at the cost of latency. To quantify the residual risk, we count *severe failures*: cases where the ViT and VLM trajectories have a large score gap but the system still fails to fall back to the VLM branch.

J Oracle Controls Across Model Families

To test whether the gains in RQ2/RQ3 are simply due to generic ensemble diversity, we also computed oracle controls across vision–vision and VLM–VLM pairings. Their gains are substantially smaller than those of VLM–vision pairing, suggesting that not all diversity is equally useful.

Table 18: Oracle controls across model families. Gains are measured relative to the best single model inside each candidate set.

Candidate set	Oracle score	Gain
ViT + ResNet	89.75	+0.87
ViT + EVA-CLIP	89.61	+0.73
ResNet + EVA-CLIP	88.69	+0.80
ViT + ResNet + EVA-CLIP	90.05	+1.17
InternVL + Qwen	91.90	+1.10
VLM + ViT	93.58	+2.78

K Qualitative Case Gallery (VLM vs. ViT vs. Expert)



Figure 7: Qualitative case gallery arranged in two columns. Each panel contains a front-camera view and a BEV visualization. Red denotes the VLM trajectory, blue denotes the ViT trajectory, and green denotes the human (expert) trajectory used as imitation-learning supervision. The left column (Cases 1–5) primarily highlights longitudinal / speed-profile differences, while the right column (Cases 6–10) highlights lateral / path and lane-level preference differences.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction accurately summarize our main claims: representation complementarity, behavior-level differences, and trajectory-level exploitation through HybridDriveVLA and DualDriveVLA.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the scope of the studied VLA paradigm and note that conclusions may not directly transfer to more entangled LLM-centric driving architectures.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: This paper does not present formal theoretical theorems or proofs. The mathematical content defines diagnostics, scores, and model components.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We describe the two-branch architecture, feature probes, SAE/CKA/CCA analysis, gating setup, trajectory scorer, candidate construction, and NAVSIM evaluation protocol.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: Code is not released at submission time because it is still being cleaned and anonymized. We plan to release the codebase after the review process.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We specify the NAVSIM setting, train/test protocol, model variants, feature extraction points, metrics, and main implementation details, with additional details in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We include stability-aware win counting and report behavioral/safety diagnostics with standard deviations over multiple random seeds in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We report the inference hardware and latency for the VLM baseline and DualDriveVLA. The appendix should also list the training GPU setup and approximate runtime for the main experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses public autonomous-driving benchmarks and offline evaluation. It does not involve human subjects, private data, or real-world vehicle deployment.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

Justification: We do not include a dedicated broader-impact section. Real-world societal and safety impacts of autonomous-driving deployment require separate validation beyond this offline benchmark study.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: We cite the datasets, evaluators, pretrained backbones, VLMs, and baseline methods used in the experiments. We should list asset versions and licenses in the appendix or code release.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We cite the datasets, evaluators, pretrained backbones, VLMs, and baseline methods used in the experiments.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [No]

    Justification: We do not release new assets at submission time. The planned code release will include documentation, usage instructions, and license information.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing, user studies, or human-subject

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

Justification: IRB approval is not applicable because the work does not involve human subjects or private personal data.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: VLMs are a core object of study and are explicitly described as frozen language-grounded visual representation sources in the planning framework.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.