VAULT: ONE-STEP LATENT GENERATION WITH POSITIVE-ANCHORED REWARDS FOR AUTONOMOUS DRIVING

Anonymous authors
Paper under double-blind review

ABSTRACT

End-to-end autonomous driving must balance multimodal maneuver generation against real-time inference constraints. Diffusion planners capture diverse behaviors, but their iterative denoising incurs considerable inference cost in safety-critical deployment, while one-step alternatives still imitate only the single expert demonstration, so neither family certifies both the diversity and the quality of its outputs. To address this, we propose Vault, a framework that couples one-step latent generation with sample-based reinforcement learning guidance. Built on a V-JEPA 2.1 visual representation pretrained on the full NAVSIM training imagery, Vault generates trajectory candidates in a single forward pass by drifting in a VAE latent space. During training, a reward-gated positive pool collects the maximum-scoring trajectories that the model itself discovers under the official evaluator and anchors the drift targets to these certified samples. This closes a reinforcement-learning loop purely through sample-based target construction, without policy gradients, learned reward models, or extra interaction: the generator imitates the frontier of its own certified successes, so quality and multimodality improve together under a single fixed sampling configuration. A learned scorer that predicts the official score and its sub-metrics then selects the executed trajectory, and all training-only modules are discarded at inference. On the NAVSIM benchmarks, Vault achieves state-of-the-art performance at real-time inference cost, reaching 94.6 PDMS on v1 and 91.2 EPDMS on v2.

1 INTRODUCTION

End-to-end (E2E) autonomous driving maps sensor inputs directly to trajectories (Hu et al., 2023; Jiang et al., 2023a; Li et al., 2024a), yet urban driving is inherently multimodal (Jiang et al., 2023b): at a crowded unsignalized intersection, yielding, proceeding, and merging into cross-traffic are all valid, and deterministic regression, the default in most E2E planners, averages across these modes into physically implausible trajectories that straddle incompatible maneuvers. Diffusion planners capture multimodality by iterative denoising (Zhong et al., 2023; Zheng et al., 2025a; Xing et al., 2025), but their latency, tens to hundreds of forward passes per prediction, exceeds real-time budgets (Ho et al., 2020; Song et al., 2021), and accelerated samplers (Song et al., 2023; Liu et al., 2022; Lipman et al., 2023) sacrifice trajectory quality for speed. A closer look at the denoising chain shows that most of it performs geometric refinement rather than the semantic shift from the unconditional prior to the scene-conditional distribution. The theory of Generative Modeling via Drifting (Deng et al., 2026) shows that once geometric decoding is offloaded to a separate module, the semantic shift takes a single step.

One-step generation removes the latency bottleneck but leaves supervision untouched: whatever the sampler architecture, the targets remain static. Diffusion planners regress toward the single expert demonstration of each scene, so diversity must be manufactured by the noise initialization. One-step latent planners (Zhang et al., 2025) compress the denoising chain yet inherit the same demonstration targets. Multimodality is thus borrowed from the sampling configuration rather than taught by supervision. Covering distinct maneuvers requires sweeping guidance strengths or candidate counts at inference. Moreover, the endpoint region of generation is never certified: nothing in the objective

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

guarantees that the diverse modes are safe, drivable, or rewarded, and a static target cannot improve as the policy improves, leaving imitation quality and distributional coverage in permanent tension.

We propose Vault, which closes this gap by routing the official planning reward into the construction of the generation targets. During training, a reward-gated positive pool incrementally collects the trajectories that the model itself discovers and that the official evaluation certifies with a perfect score. The drift targets are anchored to these certified positives alongside the expert demonstration. Because every anchor is flawless by construction while the pool spans the distinct maneuvers a scene admits, the matched distribution is simultaneously high-quality and multimodal, and a single fixed sampling configuration suffices at inference. Vault realizes this design with a V-JEPA 2.1-based backbone self-pretrained on the full navtrain imagery (Dauner et al., 2024), a one-step Vault Decoder, and a Vault Scorer that selects the executable candidate by predicting the official score and its sub-metrics. The pool and the VAE encoder used to construct targets are discarded at inference, leaving neither iterative sampling nor auxiliary stages in the deployed model.

Our main contributions are as follows:

- **A one-step trajectory generation method.** A scene-conditioned decoder produces all trajectory proposals in a single forward pass, trained by kernel-weighted attraction to anchors and repulsion among candidates, with geometric decoding offloaded to a frozen PCA basis.
- **A self-evolving, sample-based reward scheme.** A reward-gated pool collects the maximum-scoring trajectories the model itself discovers under the official evaluator and anchors the drift targets to them, so supervision evolves with the policy through samples alone, without policy gradients, learned reward models, or extra interaction.
- **State-of-the-art NAVSIM results with fast, large-batch generation.** Under a single fixed sampling configuration, Vault surpasses all compared planners on NAVSIM v1 and v2, and its decoder emits the entire candidate set within milliseconds.

2 Related Work

E2E driving and visual representations. E2E driving has evolved from modular pipelines to unified architectures that plan directly from camera input (Hu et al., 2023; Jiang et al., 2023a; 2026), with recent systems introducing trajectory-vocabulary prediction, diffusion decoding, and vision-language-action modeling (Li et al., 2025a; Liao et al., 2025; Zhou et al., 2026). Joint-embedding pretraining has likewise been adapted to driving (Wang et al., 2026). Vault adopts the V-JEPA 2.1 architecture from general video modeling (Assran et al., 2025; Mur-Labadia et al., 2026), pretrains it on four-view navtrain imagery, and keeps it trainable with the planner.

Generative planners. Diffusion and flow-based planners model multiple maneuvers by iterative sampling (Jiang et al., 2023b; Zhong et al., 2023; Zheng et al., 2025a; Xing et al., 2025; Liao et al., 2025). In general generative modeling, consistency models, rectified flow, and flow matching reduce the number of sampling steps (Song et al., 2023; Liu et al., 2022; Lipman et al., 2023). Driving planners follow this direction: DiffusionDriveV2 truncates the denoising chain under reward constraints (Zou et al., 2025), while LAP denoises in a single step within a trajectory-VAE space but still regresses toward the logged demonstration (Zhang et al., 2025). Drifting (Deng et al., 2026) instead evolves the generated distribution across training iterations, so the deployed generator needs no denoising chain at inference. Vault instantiates this drift objective for planning and extends its anchor set with reward-filtered, self-generated trajectories.

Reward-guided learning and trajectory scoring. Self-imitation learning and reward-weighted regression improve policies by selectively reusing past successes (Oh et al., 2018). The positive pool carries this principle into generative planning: it is self-imitation gated by a perfect official score, so the imitated successes are precisely the trajectories that the evaluator certifies. For candidate selection, Hydra-MDP, GTRS, DriveSuprim, and DrivoR learn to rank trajectory proposals with trained scorers (Li et al., 2024a; 2025b; Yao et al., 2026; Kirby et al., 2026), and Vault adopts a Transformer scorer similar to DrivoR without its visual register tokens. Closest to our reward-guided training, CLOVER combines evaluator-filtered pseudo-experts with scorer-mediated self-distillation (Ang et al., 2026). Vault differs on both axes: its drift targets are the model's own

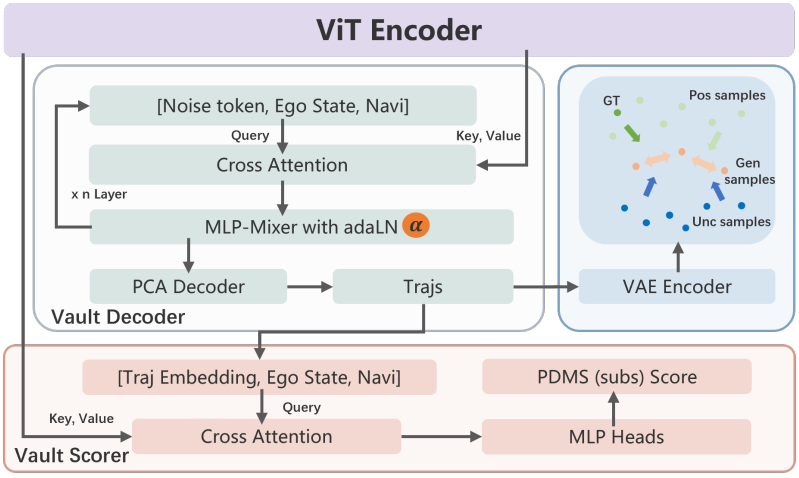


Figure 1: Overview of Vault. A V-JEPA 2.1 ViT produces visual tokens. Each noise input is grouped with ego-state and navigation tokens before visual cross-attention and PCA decoding. The scorer concatenates each trajectory embedding with the ego-state and navigation tokens and ranks candidates by predicted PDMS and sub-metrics. In the training-only inset, green denotes expert/pool anchors, blue the scene-agnostic vocabulary, and orange generated candidates. The VAE encoder and positive pool are absent at inference.

maximum-scoring trajectories rather than filtered pseudo-experts, and its scorer only ranks decoded candidates at inference rather than mediating generator training.

3 METHOD

3.1 OVERVIEW

Vault takes surround-view images, ego state, and a navigation command, and outputs trajectory proposals from which one is executed. A ViT following the V-JEPA 2.1 architecture (Assran et al., 2025; Mur-Labadia et al., 2026), pretrained on four-view navtrain imagery and trainable with the planner, produces visual tokens $\mathbf{F}_{\text{vis}}$. Velocity and acceleration form an ego-state token $\mathbf{e}_{\text{ego}}$, and the one-hot command forms $\mathbf{e}_{\text{navi}}$. The decoder groups these state and command tokens with each noise token and attends to $\mathbf{F}_{\text{vis}}$. The scorer concatenates each candidate embedding with the ego-state and navigation tokens and attends to $\mathbf{F}_{\text{vis}}$.

The *Vault Decoder* maps N noise inputs to trajectory proposals in a single forward pass, through cross-attention and α-conditioned MLP-Mixer blocks followed by a frozen PCA basis (Section 3.2). The Mixer acts within each candidate's three-token group (noise, ego state, navigation), while N is a batch-like candidate axis, so the same weights accept different candidate counts. The *Vault Scorer* embeds the proposals together with the ego-state and navigation tokens, attends to the visual tokens, and predicts PDMS and its sub-metrics. The top-ranked candidate is executed. Training also uses a frozen trajectory VAE encoder and a *positive pool* of self-generated, maximum-scoring training trajectories (Section 3.3). The expert anchor is retained independently of its score. At inference, the graph contains the visual encoder, decoder, and scorer. Figure 1 summarizes the architecture.

3.2 ONE-STEP GENERATION VIA DRIFTING

Drifting (Deng et al., 2026) updates a generator during training rather than applying a denoising chain at inference. Vault defines this update in the latent space of a trajectory VAE and decodes output waypoints through a frozen PCA basis. A generator f_θ maps standard Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ directly to a sample $\mathbf{x} = f_\theta(\boldsymbol{\epsilon})$. Its samples are regressed toward displaced copies along a

field built from the target distribution p and current generated distribution q,

$$\mathbf{V}_p^+(\mathbf{x}) = \frac{\mathbb{E}_{\mathbf{y}^+\sim p}\big[k(\mathbf{x},\mathbf{y}^+)\,(\mathbf{y}^+-\mathbf{x})\big]}{\mathbb{E}_{\mathbf{y}^+\sim p}\big[k(\mathbf{x},\mathbf{y}^+)\big]}, \qquad \mathbf{V}_q^-(\mathbf{x}) = \frac{\mathbb{E}_{\mathbf{y}^-\sim q}\big[k(\mathbf{x},\mathbf{y}^-)\,(\mathbf{y}^--\mathbf{x})\big]}{\mathbb{E}_{\mathbf{y}^-\sim q}\big[k(\mathbf{x},\mathbf{y}^-)\big]}, \tag{1}$$

with $\mathbf{V}_{p,q} = \mathbf{V}_p^+ - \mathbf{V}_q^-$ pulling $\mathbf{x}$ toward positive samples $\mathbf{y}^+$ and away from negative samples $\mathbf{y}^-$ through the kernel $k(\mathbf{x},\mathbf{y}) = \exp(-\|\mathbf{x}-\mathbf{y}\|_2/\tau)$:

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{\epsilon}}\Big[\big\|f_\theta(\boldsymbol{\epsilon}) - \mathrm{sg}\big(f_\theta(\boldsymbol{\epsilon}) + \mathbf{V}_{p,q_\theta}(f_\theta(\boldsymbol{\epsilon}))\big)\big\|^2\Big], \tag{2}$$

where $\mathrm{sg}(\cdot)$ is the stop-gradient operator. Vanishing drift is a fixed point of this update. Appendix A recaps the theory.

We construct the objective in the latent space of a trajectory VAE trained on reconstruction with an auxiliary maneuver-classification head, which encourages behavioral separation in the latent space (Appendix C compares candidate spaces). Only the frozen VAE encoder is used during planner training, and physical trajectories come from the PCA decoder described next.

Vault Decoder. Each of the N noise embeddings is paired with the ego-state and navigation embeddings, giving an array of shape $B \times N \times 3 \times D_{\mathrm{dec}}$ for batch size B and decoder feature width D_{dec}. Each three-token group attends to visual scene features and passes through interleaved cross-attention and MLP-Mixer blocks. The Mixer operates along the within-candidate three-token and feature axes, with parameters shared across N, so it never mixes candidates. Other candidates influence a proposal only through the training-time repulsion field. Guidance strength α enters through Adaptive Layer Normalization (Peebles & Xie, 2023), and the output is projected to waypoints $\{\boldsymbol{\tau}_i\}_{i=1}^N$ through a frozen PCA basis fitted on expert demonstrations.

Instantiated drift field. For each scene, decoder waypoints $\boldsymbol{\tau}_i$ pass through the frozen VAE encoder to give $\mathbf{z}_i = \mathrm{VAE}_{\mathrm{enc}}(\boldsymbol{\tau}_i)$. Gradients propagate through its operations to the decoder while its weights remain fixed. The conditional anchor set $\mathcal{A} = \{\mathbf{a}_1,\dots,\mathbf{a}_M\}$ contains the expert latent $\mathbf{z}_{\mathrm{GT}} = \mathrm{VAE}_{\mathrm{enc}}(\boldsymbol{\tau}^{\mathrm{GT}})$ and pool positives (Section 3.3). Only the latter must pass the maximum-score gate. The negatives are the other concurrently generated candidates, $\mathcal{N}_i = \{\mathbf{z}_j\}_{j\neq i}$. Euclidean latent distances, where D is the VAE latent dimension, are normalized by a stopped scene scale $S_{\mathrm{sc}} = \mathrm{sg}\big(\max\{\overline{d(\mathbf{z},\mathcal{A})}/\sqrt{D}, \varepsilon_s\}\big)$. The floor ε_s keeps this scale defined when candidates and anchors coincide. The coupled assignment is

$$r_{ij} = \frac{e^{-d(\mathbf{z}_i,\mathbf{a}_j)/(\tau_\ell S_{\mathrm{sc}})}}{\sum_{j'} e^{-d(\mathbf{z}_i,\mathbf{a}_{j'})/(\tau_\ell S_{\mathrm{sc}})}}, \qquad c_{ij} = \frac{e^{-d(\mathbf{z}_i,\mathbf{a}_j)/(\tau_\ell S_{\mathrm{sc}})}}{\sum_{i'} e^{-d(\mathbf{z}_{i'},\mathbf{a}_j)/(\tau_\ell S_{\mathrm{sc}})}}, \qquad w_{ij}^+ = \frac{\sqrt{r_{ij}\,c_{ij}}}{\sum_{j'}\sqrt{r_{ij'}\,c_{ij'}}}, \tag{3}$$

with w_{im}^- defined identically on $\mathcal{N}_i$ after masking the self-match $m = i$. Combining the row normalization over anchors with the column normalization over candidates, each side becomes a kernel-weighted mean,

$$\mathbf{A}_i = \sum_{j=1}^{M} w_{ij}^+\,\mathbf{a}_j, \qquad \mathbf{R}_i = \sum_{m\neq i} w_{im}^-\,\mathbf{z}_m. \tag{4}$$

The assignment permits several candidates to share an anchor while the repulsive term encourages separation. The kernel is evaluated at three bandwidths $\{\tau_\ell\} = \{1,2,4\}$ and the displacements are summed. Appendix B relates these finite-sample, bidirectionally normalized fields to Eq. equation 1.

The total field mixes a *conditional* branch using the expert and pool anchors for the current scene with an *unconditional* branch using a scene-agnostic vocabulary of maneuver prototypes obtained by clustering all logged expert trajectories within left-turn, straight, right-turn, and stop classes, from which K prototypes are drawn each iteration (Appendix B.7):

$$\begin{aligned} \mathbf{V}_i^{\mathrm{cond}} &= \sum_\ell \big(\mathbf{A}_{i,\ell} - \mathbf{R}_{i,\ell}\big), \qquad \mathbf{V}_i^{\mathrm{unc}} = \sum_\ell \big(\bar{\mathbf{A}}_{i,\ell} - \mathbf{R}_{i,\ell}\big), \\ \mathbf{V}_i &= \mathbf{V}_i^{\mathrm{unc}} + \alpha_i\big(\mathbf{V}_i^{\mathrm{cond}} - \mathbf{V}_i^{\mathrm{unc}}\big), \qquad \mathcal{L}_{\mathrm{drift}} = \frac{1}{ND}\sum_{i=1}^{N}\Big\|\mathbf{z}_i - \mathrm{sg}\big(\mathbf{z}_i + \mathbf{V}_i/\lambda_i\big)\Big\|_2^2, \end{aligned} \tag{5}$$

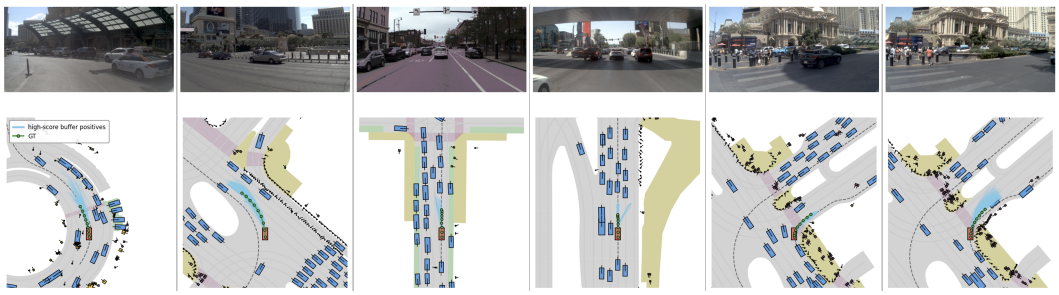


Figure 2: Reward-gated positive pool examples from six training scenes (front view above, BEV below). Green dotted marks the logged expert, and light blue marks self-generated pool trajectories admitted with the maximum evaluator score on those training scenes. Some retained branches extend farther along the route than the logged expert.

where $\bar{\mathbf{A}}_{i,\ell}$ uses the scene-agnostic vocabulary instead of scene anchors and $\lambda_i = \mathrm{sg}\big(\max\{\sqrt{\|\mathbf{V}\|_2^2/D}, \varepsilon\}\big)$ normalizes the drift step, with $\varepsilon > 0$ preventing division by zero. The stopped target gives a surrogate gradient proportional to $-\mathbf{V}_i/\lambda_i$ with respect to $\mathbf{z}_i$. At $\alpha = 1$ the unconditional attraction cancels. Appendix B analyzes the resulting fixed-point condition.

The α family. Training samples $\alpha_i \sim \mathcal{U}[0, 2]$ per candidate: $\alpha = 0$ uses the scene-agnostic vocabulary, $\alpha = 1$ uses scene anchors, and $\alpha > 1$ extrapolates the conditional drift direction away from the unconditional one (Appendix A). Inference fixes $\alpha = 1$ and uses one conditional forward pass.

3.3 Reward-Gated Positive Pool

During training, a pool $\mathcal{Q}$ stores latent codes of self-generated, fully feasible trajectories, each admitted by attaining the maximum score under the official evaluator on its training scene. At the end of each epoch, every newly qualifying candidate is encoded by the frozen VAE encoder and admitted to the pool, and a fixed number of stored positives is drawn per scene and iteration. This expands the supervised anchor set beyond the single logged expert. A maximum-scoring trajectory attains maximal ego progress together with maximal safety and comfort sub-scores, so the generator can imitate trajectories that advance farther than a lower-progress demonstration. Figure 2 shows examples, while the pool ablation in Section 4.4 assesses the system-level effect.

This is reward-filtered self-imitation: the evaluator supplies a training-only admission signal, without differentiating through the metric. Unlike a learned reward model, the gate directly uses the official score of the target benchmark, and a planner trained for a given benchmark maintains its own pool.

3.4 Vault Scorer

Vault follows the generate-and-score paradigm (Li et al., 2024a; Yao et al., 2026), adopting the Transformer-based scorer of DrivoR (Kirby et al., 2026). Each decoded trajectory $\boldsymbol{\tau}_i$ is embedded by a lightweight trajectory encoder, and the embedding is concatenated with the ego-state and navigation tokens $\mathbf{e}_{\text{ego}}$ and $\mathbf{e}_{\text{navi}}$. These concatenated features serve as queries in a Transformer decoder whose keys and values are the same visual tokens that the Vault Decoder attends to, and MLP heads map each output to an overall score S_i and sub-scores $\{S_i^{(m)}\}$ estimating the PDMS and its sub-metrics (e.g., no at-fault collision, drivable area compliance, ego progress, time-to-collision, comfort).

Training labels come from the official scorer applied to the generated trajectories: y_i is the PDMS of candidate i and $y_i^{(m)}$ its sub-metrics. A pairwise hinge loss encourages score ordering, while a multi-task regression term fits the overall and component scores, $\mathcal{L}_{\text{scorer}} = \lambda_{\text{rank}}\mathcal{L}_{\text{rank}} + \lambda_{\text{mse}}\mathcal{L}_{\text{mse}}$ (Appendix D). At inference, the highest predicted overall score determines the executed candidate.

4 EXPERIMENTS

We report NAVSIM v1 and v2 comparisons (Section 4.2), a generator swap and candidate-oracle analysis (Section 4.3), a pool ablation (Section 4.4), component latency (Section 4.5), and a qualitative trajectory analysis (Section 4.6). The generator swap matches candidate count but changes the decoder and its training objective.

4.1 EXPERIMENTAL SETUP

Benchmark and metrics. NAVSIM (Dauner et al., 2024) evaluates planned trajectories in non-reactive pseudo-simulation against logged scenes. We train on navtrain (1,192 scenarios, about 85k frames) and evaluate on navtest (about 12k scenarios), using four of eight camera views. The v1 PDM score (PDMS) combines collision avoidance (NC), drivable-area compliance (DAC), ego progress (EP), time-to-collision (TTC), and comfort (C). The v2 benchmark (Cao et al., 2025) adds compliance and comfort terms to the score (EPDMS), and the v2 planner is trained separately from the v1 planner (Section 4.1). The original and corrected v2 evaluators have different human-reference filtering behavior, and scores computed with them should be compared within the same column (Appendix E).

Training details. Training has a pre-training stage followed by the main loop. Stage 0 pre-trains the ViT on navtrain, fits the trajectory VAE and PCA basis on expert demonstrations, and builds the scene-agnostic vocabulary of maneuver prototypes for the unconditional branch (Appendix B.7). The VAE encoder, PCA basis, and vocabulary are frozen thereafter, while the ViT remains trainable. The v1 and v2 planners are trained separately and share only the navtrain-pretrained ViT. Each iteration samples a scene, generates N candidates with one sampled $\alpha_i \sim \mathcal{U}[0, 2]$ per candidate, and assembles conditional anchors from the expert and up to $M - 1$ pool positives. The planner is updated with Eq. equation 5 and the scorer with labels from the run's official evaluator. Every T iterations, candidates are regenerated on training scenes and maximum-scoring latent codes are added to $\mathcal{Q}$. At inference, $\alpha = 1$ cancels the scene-agnostic branch, and the vocabulary, pool, and VAE are not evaluated. The reported navtest scores select one of $N = 1024$ proposals with the learned scorer, and at most $M = 32$ scene anchors are used during training, subject to pool availability.

4.2 COMPARISON WITH THE STATE OF THE ART

NAVSIM v1. With 1024 proposals and learned-scorer selection, Vault reaches 94.6 PDMS on v1 navtest (Table 1), 0.1 above CLOVER's reported 94.5 (Ang et al., 2026) and 0.9 above the DrivoR run without SimScale augmentation (Kirby et al., 2026). Across five random seeds, v1 PDMS ranges from 94.52 to 94.60, all exceeding CLOVER's reported 94.5, and the reported run uses seed 1234, whose score is closest to the five-seed average. We use that DrivoR variant because Vault also has no simulated-data augmentation. Vault's EP is 92.9, above both the 91.7 of CLOVER and the 87.5 human-reference EP, while its comfort is 99.0 versus 99.9 for the human reference. Thus the selected planner makes more evaluator-measured progress than the logged human on average, with a comfort trade-off, and the human reference still has a higher aggregate PDMS of 94.8. The pool ablation below tests which part of this behavior depends on reward-gated training anchors.

NAVSIM v2. On v2 navtest, Vault obtains 88.3 EPDMS* under the original evaluator and 91.2 EPDMS under the corrected evaluator (Table 2). The corrected result exceeds the paper-reported scores of SparseDriveV2 (90.1), CLOVER (90.4), and SUV (91.0) (Sun et al., 2026; Ang et al., 2026; Yuan et al., 2026). The 2.9-point difference between Vault's evaluator versions reflects a scoring change, not better driving. Vault has the highest mean EP among the listed planners (92.1 versus 91.1 for SparseDriveV2), above the human-reference mean of 87.4. Its lower LK (96.8), HC (97.4), and EC (76.2), however, show that higher progress does not dominate every component, and the human reference scores 94.5 EPDMS overall. The v1 and v2 planners are separately trained with their own evaluator-gated pools, so the v2 score reflects its own training metric rather than cross-metric transfer.

Table 1: NAVSIM v1 navtest results (%). Prior results are transcribed from the cited papers. The DrivoR row is the version without SimScale augmentation, as Vault also uses no simulated augmentation. Human is a reference, not a competing planner. Vault, its diffusion-decoder replacement, and the pool ablations each use 1024 proposals ranked by the learned scorer. Oracle rows select with the official evaluator and are analysis upper bounds.

Method	NC ↑	DAC ↑	EP ↑	Comfort ↑	TTC ↑	PDMS ↑
Human reference (Dauner et al., 2024)	100.0	100.0	87.5	99.9	100.0	94.8
World4Drive (Zheng et al., 2025b)	97.4	94.3	79.9	100.0	92.8	85.1
VADv2 (Jiang et al., 2026)	98.1	94.8	80.6	100.0	94.3	86.2
DiffusionDrive (Liao et al., 2025)	98.2	96.2	82.2	100.0	94.7	88.1
AutoVLA (Zhou et al., 2026)	98.4	95.6	81.9	99.9	98.0	89.1
GoalFlow (Xing et al., 2025)	98.4	98.3	85.0	100.0	94.6	90.3
ReCogDrive (Xiong et al., 2026)	97.9	97.3	87.3	100.0	94.9	90.8
Hydra-MDP++ (Li et al., 2025a)	98.6	98.6	85.7	100.0	95.1	91.0
iPAD (Guo et al., 2026)	98.6	98.3	88.0	100.0	94.9	91.7
Drive-JEPA (Wang et al., 2026)	99.1	98.2	91.4	99.9	96.2	93.7
DriveSuprim (Yao et al., 2026)	98.6	98.6	91.3	100.0	95.5	93.5
DrivoR (Kirby et al., 2026)	99.0	98.9	90.0	100.0	96.7	93.7
CLOVER (Ang et al., 2026)	99.1	99.0	91.7	100.0	96.9	94.5
Vault (ours)	99.0	98.8	92.9	99.0	96.4	**94.6**
Vault with diffusion decoder	99.1	98.6	86.5	100.0	96.5	92.0
Vault without positive pool	98.7	98.8	91.8	99.9	95.9	93.9
Vault with single positive anchor	98.3	98.7	93.0	99.3	94.7	93.8
Vault oracle best-of-64	99.7	99.8	98.1	99.9	98.6	98.4
Vault oracle best-of-256	99.8	99.8	98.5	100.0	99.4	99.0
Vault oracle best-of-1024	99.9	99.9	98.7	100.0	99.7	99.3

Table 2: NAVSIM v2 navtest results (%). EPDMS* uses the original evaluator and EPDMS the corrected evaluator. Sub-metrics are transcribed as reported by the cited sources, which do not always specify their evaluator version for each component. Dashes denote unavailable scores, and the human row is shown for reference.

Method	NC	DAC	DDC	TLC	EP	TTC	LK	HC	EC	EPDMS*	EPDMS
Human reference (Tian et al., 2026; Yuan et al., 2026)	100.0	100.0	99.8	100.0	87.4	100.0	100.0	98.1	90.1	90.3	94.5
Ego Status MLP (Li et al., 2024b)	93.1	77.9	92.7	99.6	86.0	91.5	89.4	98.3	85.4	64.0	–
Transfuser (Chitta et al., 2022)	96.9	89.9	97.8	99.7	87.1	95.4	92.7	98.3	87.2	76.7	–
ARTEMIS (Feng et al., 2025)	98.3	95.1	98.6	99.8	81.5	97.4	96.5	98.3	98.3	83.1	–
DiffusionDrive (Liao et al., 2025; Ye et al., 2026)	98.2	96.2	99.5	99.8	87.4	97.3	96.9	98.4	87.7	84.7	88.2
DiffusionDriveV2 (Zou et al., 2025)	97.7	96.6	99.2	99.8	88.9	97.2	96.0	97.8	91.0	85.5	87.5
Hydra-MDP++ (Li et al., 2025a)	98.4	98.0	99.4	99.8	87.5	97.7	95.3	98.3	77.4	85.1	–
DriveSuprim (Yao et al., 2026)	97.8	97.9	99.5	99.9	90.6	97.1	96.6	98.3	77.9	86.0	–
SparseDriveV2 (Sun et al., 2026)	98.1	98.1	99.6	99.8	91.1	97.3	96.9	98.2	78.4	86.7	90.1
CLOVER (Ang et al., 2026)	99.4	99.3	99.5	99.8	86.9	98.9	95.2	98.2	75.5	87.2	90.4
SUV (Yuan et al., 2026)	99.1	97.8	99.7	99.8	87.8	98.7	98.1	98.4	88.3	–	91.0
Vault (ours)	98.9	98.9	99.7	99.7	92.1	98.3	96.8	97.4	76.2	**88.3**	**91.2**

4.3 One-Step Drift vs. Iterative Denoising

We compare the one-step generator with a diffusion-decoder replacement and analyze the candidate set separately from the learned selection rule. The reported scores use the same $N = 1024$ candidate budget and learned scorer. The swap changes the decoder and training objective, so it does not by itself isolate the drift objective.

Generator swap at matched candidate count. With the same visual backbone, scorer, training scenes, and 1024-candidate selection budget, the diffusion decoder reaches 92.0 PDMS, versus 94.6 for Vault. Its ego progress is markedly lower, 86.5 versus 92.9, while its comfort is higher at 100.0 versus 99.0 and its safety sub-metrics are comparable, with NC 99.1 versus 99.0, DAC 98.6 versus 98.8, and TTC 96.5 versus 96.4. The diffusion replacement thus trades route progress for comfort, yielding a 2.6-point lower aggregate in this system comparison.

Oracle best-of-N. Because the Mixer operates within each candidate rather than across candidates, the same decoder accepts different numbers of noise inputs. Separate forward passes with $N = 64$, 256, and 1024 generate the three candidate sets without changing weights. Choosing the highest officially scored candidate per navtest scene yields 98.4, 99.0, and 99.3 PDMS for the full model, and 97.2, 97.8, and 98.0 for the no-pool decoder anchored only to the logged expert (Section 4.4). The generated trajectories are thus already sufficiently good, reaching 99.3 PDMS under oracle selection at $N = 1024$, and the full model's ceiling exceeds its no-pool counterpart at every N, confirming that the positive pool raises the upper bound of generation quality itself.

4.4 Ablation: the Reward-Gated Pool

The "without positive pool" variant keeps the backbone, scorer, and 1024-candidate selection configuration, while using only the logged expert as its conditional anchor. A pool trajectory must score PDMS $= 1$ on its training scene, which entails maximal v1 EP as well as the other score components, and when the logged expert has EP < 1, a full-score pool entry supplies a higher-EP target. Removing the pool lowers PDMS from 94.6 to 93.9. Among the v1 sub-scores, the largest positive change with the pool is EP, from 91.8 to 92.9. TTC rises from 95.9 to 96.4, NC from 98.7 to 99.0, DAC stays at 98.8, and comfort falls from 99.9 to 99.0. This pattern is consistent with more decisive, progress-oriented planning while maintaining high measured safety scores.

A complementary variant disables the pool and instead replaces the logged expert anchor with a single high-scoring stored positive. Anchored to this one sample, the planner pushes EP to 93.0, above the full model's 92.9, but its NC falls to 98.3 and its TTC to 94.7, yielding 93.8 PDMS. A single positive target thus supplies high-progress supervision yet is unstable, frequently producing safety violations, and neither it nor the logged expert alone matches the full pool. Anchoring the drift to a diverse set of certified positives appears to be what stabilizes training.

The aggregate PDMS is nonlinear, so these sub-score changes do not identify a precise causal share of its 0.7-point gain. A paired pool-versus-expert comparison and a pool-size sweep would provide a more direct test of this mechanism.

4.5 Inference Efficiency

On one RTX 4090, end-to-end inference at the evaluated $N = 1024$ takes 91.4 ms, of which 76.3 ms is the V-JEPA encoder on sixteen 384×672 frames, 5.4 ms the decoder, and approximately 9 ms the scorer, so the encoder dominates this budget. Candidate count contributes only marginally, and reducing N from 1024 to 64 shortens end-to-end inference to 82.8 ms. The reported configuration thus runs at about 11 scenes per second, meeting the 10 Hz planning requirement.

4.6 Trajectory Generation Analysis

This section presents the trajectories that the deployed model generates and selects on navtest (Figure 3). We show planning results on six scenes, covering a left turn, straight driving on a narrow road, multi-lane traffic with multiple vehicles, and a right turn on a narrow road. In every scene the entire 1024-candidate set comes from a single forward pass, and its spread adapts to the scene: at decision points such as intersections and turns the fan widens into distinct maneuvers, while on constrained straight segments it contracts to a narrow band around the lane center. Diversity thus appears exactly where the scene admits multiple maneuvers, rather than being spent uniformly across all scenes.

In all of these scenes, the model generates and selects reasonable trajectories directly from the visual input. On the narrow roads, every generated trajectory stays within the road boundaries, confirming the feasibility of the generated trajectories and indicating that the aggressive full-score positives in the pool do not compromise feasibility. In the two middle multi-vehicle straight scenes, guided by the pool, the model generates right-side overtaking trajectories (the two middle panels of Figure 3) and the scorer selects them, while the logged expert continues following the lead vehicle in both scenes. The source of this mode is visible in the pool itself: among the self-generated positives admitted with the maximum evaluator score (Figure 2) are branches that overtake the lead vehicle and extend farther along the route than the logged expert. Anchoring the drift targets to these certified samples supplies the overtaking mode that the logged demonstration alone does not offer in such

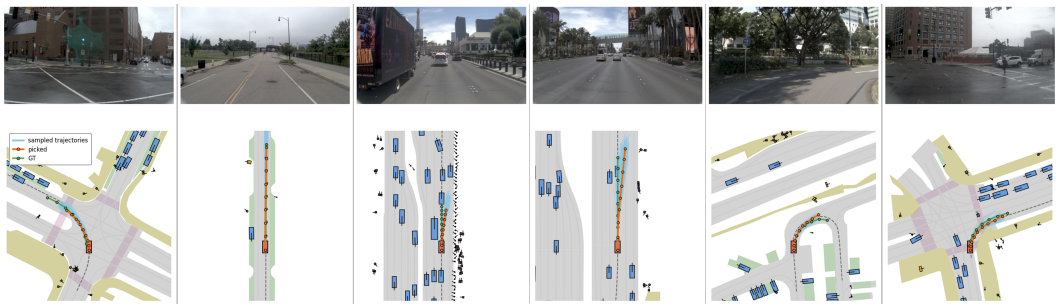


Figure 3: Generation and selection examples on navtest at $N = 1024$ (front view above, BEV below). The light-blue fan marks the 1024 candidates generated in a single forward pass, orange marks the scorer selection, and green marks the logged human trajectory.

scenes, and the executed maneuver thereby extends beyond the expert toward a higher-progress alternative that the official evaluator certifies. In the remaining scenes the selected trajectory coincides with the logged human trajectory or deviates only by a small lateral offset through the turns, and the few candidates that leave the roadway are never selected. The self-evolving pool thus effectively guides the generation of diverse, high-quality trajectories, which underlies the EP gains observed in the pool ablation (Section 4.4).

5 Conclusion

We presented Vault, an end-to-end planner that couples one-step trajectory generation with a self-evolving, sample-based reward scheme. The Vault Decoder produces all trajectory proposals in a single forward pass, trained by kernel-weighted attraction to anchors and repulsion among candidates in a VAE latent space, with geometric decoding offloaded to a frozen PCA basis. A reward-gated positive pool collects the maximum-scoring trajectories that the model itself discovers under the official evaluator and anchors the drift targets to them, so supervision evolves with the policy through samples alone, without policy gradients, learned reward models, or extra interaction; the generator imitates the frontier of its own certified successes, and quality and multimodality improve together under a single fixed sampling configuration. A learned scorer that predicts the official score selects the executed candidate, and all training-only modules are discarded at inference, leaving neither iterative sampling nor auxiliary stages in the deployed model. On NAVSIM, Vault reaches 94.6 PDMS on v1 and 91.2 corrected EPDMS on v2, with end-to-end inference of 91.4 ms at $N = 1024$ meeting the 10 Hz planning requirement; oracle selection over the same generated sets reaches 99.3 PDMS, and the pool ablation attributes both the deployed gains and this generation ceiling to the reward-gated anchors. Testing on broader datasets beyond NAVSIM remains future work.

AI use statement

The authors used a generative AI assistant for manuscript editing, consistency and formatting checks, and simulated peer review. The assistant also helped narrow the scope of several claims. They did not use generative AI for research ideation, code, experiments, or figures. The authors reviewed the AI-assisted text and take responsibility for the paper's claims, citations, numerical results, and final content.

Ethics statement

This study evaluates an offline autonomous-driving planner on NAVSIM. Benchmark scores do not establish safe behavior in real traffic, and deployment would require additional validation, including rare-event and cross-domain testing.

Reproducibility statement

Sections 3–4.1 and Appendices A–F describe the architecture, training objective, splits, evaluator versions, reported analyses, and implementation details.

References

Sining Ang, Yuguang Yang, Canyu Chen, and Yan Wang. Clover: Closed-loop value estimation and ranking for end-to-end autonomous driving planning. *arXiv preprint arXiv:2605.15120*, 2026.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

Wei Cao, Marcel Hallgarten, Tianyu Li, Daniel Dauner, Xunjiang Gu, Caojun Wang, Yakov Miron, Marco Aiello, Hongyang Li, Igor Gilitschenski, Boris Ivanovic, Marco Pavone, Andreas Geiger, and Kashyap Chitta. Pseudo-simulation for autonomous driving. In *Conference on Robot Learning (CoRL)*, 2025.

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.

Daniel Dauner, Marcel Hallgarten, Tianyu Li, Xinshuo Weng, Zhiyu Huang, Zetong Yang, Hongyang Li, Igor Gilitschenski, Boris Ivanovic, Marco Pavone, et al. Navsim: Data-driven non-reactive autonomous vehicle simulation and benchmarking. *Advances in Neural Information Processing Systems*, 37:28706–28719, 2024.

Mingyang Deng, He Li, Tianhong Li, Yilun Du, and Kaiming He. Generative modeling via drifting. *arXiv preprint arXiv:2602.04770*, 2026.

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. *IEEE Robotics and Automation Letters*, 11(1):226–233, 2025.

Ke Guo, Haochen Liu, Xiaojun Wu, Jia Pan, and Chen Lv. ipad: Iterative proposal-centric end-to-end autonomous driving. *IEEE Robotics and Automation Letters*, 2026.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 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 *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17853–17862. IEEE, 2023.

Bo Jiang, Shaoyu Chen, Qing Xu, Bencheng Liao, Jiajie Chen, Helong Zhou, Qian Zhang, Wenyu Liu, Chang Huang, and Xinggang Wang. Vad: Vectorized scene representation for efficient autonomous driving. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 8306–8316. IEEE, 2023a.

Bo Jiang, Shaoyu Chen, Hao Gao, Bencheng Liao, Qian Zhang, Wenyu Liu, and Xinggang Wang. Vadv2: End-to-end vectorized autonomous driving via probabilistic planning. In *International Conference on Learning Representations*, volume 2026, pp. 68886–68900, 2026.

Chiyu Jiang, Andre Cornman, Cheolho Park, Benjamin Sapp, Yin Zhou, Dragomir Anguelov, et al. Motiondiffuser: Controllable multi-agent motion prediction using diffusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9644–9653, 2023b.

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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 32058–32069, 2026.

Kailin Li, Zhenxin Li, Shiyi Lan, Yuan Xie, Zhizhong Zhang, Jiayi Liu, Zuxuan Wu, Zhiding Yu, and Jose M Alvarez. Hydra-mdp++: Advancing end-to-end driving via expert-guided hydra-distillation. *arXiv preprint arXiv:2503.12820*, 2025a.

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*, 2024a.

Zhenxin Li, Wenhao Yao, Zi Wang, Xinglong Sun, Joshua Chen, Nadine Chang, Maying Shen, Zuxuan Wu, Shiyi Lan, and Jose M Alvarez. Generalized trajectory scoring for end-to-end multimodal planning. *arXiv preprint arXiv:2506.06664*, 2025b.

Zhiqi Li, Zhiding Yu, Shiyi Lan, Jiahan Li, Jan Kautz, Tong Lu, and Jose M Alvarez. Is ego status all you need for open-loop end-to-end autonomous driving? In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14864–14873. IEEE, 2024b.

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. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12037–12047. IEEE, 2025.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

Lorenzo Mur-Labadia, Matthew Muckley, Amir Bar, Mido Assran, Koustuv Sinha, Mike Rabbat, Yann LeCun, and Nicolas Ballas. V-jepa 2.1: Unlocking dense features in video self-supervised learning. In *European Conference on Computer Vision*, pp. 671–689. Springer, 2026.

Junhyuk Oh, Yijie Guo, Satinder Singh, and Honglak Lee. Self-imitation learning. In *International conference on machine learning*, pp. 3878–3887. PMLR, 2018.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4172–4182. IEEE, 2023.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=St1giarCHLP.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023.

Wenchao Sun, Xuewu Lin, Keyu Chen, Zixiang Pei, Xiang Li, Yining Shi, and Sifa Zheng. Sparsedrivev2: Scoring is all you need for end-to-end autonomous driving. In *European Conference on Computer Vision*, pp. 446–463. Springer, 2026.

Haochen Tian, Tianyu Li, Haochen Liu, Jiazhi Yang, Yihang Qiu, Guang Li, Junli Wang, Yinfeng Gao, Zhang Zhang, Liang Wang, et al. Simscale: Learning to drive via real-world simulation at scale. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 36365–36374, 2026.

Linhan Wang, Zichong Yang, Chen Bai, Guoxiang Zhang, Xiaotong Liu, Xiaoyin Zheng, Xiao-Xiao Long, Chang-Tien Lu, and Cheng Lu. Drive-jepa: Video jepa meets multimodal trajectory distillation for end-to-end driving. *arXiv preprint arXiv:2601.22032*, 2026.

Zebin Xing, Xingyu Zhang, Yang Hu, Bo Jiang, Tong He, Qian Zhang, Xiaoxiao Long, and Wei Yin. Goalflow: Goal-driven flow matching for multimodal trajectories generation in end-to-end autonomous driving. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1602–1611. IEEE, 2025.

Kaixin Xiong, Xiangyu Guo, Fang Li, Sixu Yan, Gangwei Xu, Lijun Zhou, Long Chen, Haiyang Sun, Bing Wang, Kun Ma, et al. Recogdrive: A reinforced cognitive framework for end-to-end autonomous driving. In *International Conference on Learning Representations*, volume 2026, pp. 157518–157556, 2026.

Wenhao Yao, Zhenxin Li, Shiyi Lan, Zi Wang, Xinglong Sun, Jose M Alvarez, and Zuxuan Wu. Drivesuprim: Towards precise trajectory selection for end-to-end planning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 11910–11918, 2026.

Yuqi Ye, Zijian Zhang, Junhong Lin, Shangkun Sun, Changhao Peng, and Wei Gao. Autodrive-p3: Unified chain of perception-prediction-planning thought via reinforcement fine-tuning. *arXiv preprint arXiv:2603.28116*, 2026.

Yibo Yuan, Jiacheng Fu, Jiangtong Zhu, Yi Li, Jianhua Han, Meng Tian, Zhuohan Liu, Zhiwei Xiong, Hang Xu, Jianwu Fang, et al. Suv: Future scene understanding as video generation for end-to-end driving. *arXiv preprint arXiv:2608.03084*, 2026.

Jinhao Zhang, Wenlong Xia, Zhexuan Zhou, Haoming Song, Youmin Gong, and Jie Mei. Lap: Fast latent diffusion planner for autonomous driving. *arXiv preprint arXiv:2512.00470*, 2025.

Yinan Zheng, Ruiming Liang, Kexin Zheng, Jinliang Zheng, Liyuan Mao, Jianxiong Li, Weihao Gu, Rui Ai, Shengbo Li, Xianyuan Zhan, et al. Diffusion-based planning for autonomous driving with flexible guidance. In *International conference on learning representations*, volume 2025, pp. 37207–37227, 2025a.

Yupeng Zheng, Pengxuan Yang, Zebin Xing, Qichao Zhang, Yuhang Zheng, Yinfeng Gao, Pengfei Li, Teng Zhang, Zhongpu Xia, Peng Jia, et al. World4drive: End-to-end autonomous driving via intention-aware physical latent world model. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 28632–28642. IEEE, 2025b.

Ziyuan Zhong, Davis Rempe, Yuxiao Chen, Boris Ivanovic, Yulong Cao, Danfei Xu, Marco Pavone, and Baishakhi Ray. Language-guided traffic simulation via scene-level diffusion. In *Conference on robot learning*, pp. 144–177. PMLR, 2023.

Zewei Zhou, Tianhui Cai, Seth 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. *Advances in Neural Information Processing Systems*, 38: 27920–27956, 2026.

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 Background: Generative Modeling via Drifting

This section recaps the theory of Generative Modeling via Drifting (Deng et al., 2026) to the extent needed for Section 3.2. Appendix B details Vault's instantiation and its departures from the reference implementation.

Pushforward at training time. A network f_θ maps noise $\boldsymbol{\epsilon} \sim p_{\boldsymbol{\epsilon}}$ to a sample $\mathbf{x} = f_\theta(\boldsymbol{\epsilon})$. The distribution of the output is the pushforward $q_\theta = [f_\theta]_{\#} p_{\boldsymbol{\epsilon}}$. Diffusion and flow models evolve q iteratively at inference time. Drifting instead exploits the fact that SGD training already produces a sequence of models $\{f_i\}$, and lets the training iterations themselves evolve the pushforwards $q_i = [f_i]_{\#} p_{\boldsymbol{\epsilon}}$, so that the deployed generator needs a single forward pass.

Drifting field and equilibrium. The evolution is governed by a field $\mathbf{V}_{p,q}$ prescribing the update $\mathbf{x}_{i+1} = \mathbf{x}_i + \mathbf{V}_{p,q_i}(\mathbf{x}_i)$ across training iterations. For an anti-symmetric field, $\mathbf{V}_{p,q} = -\mathbf{V}_{q,p}$ implies $\mathbf{V}_{p,p} = 0$. The converse need not hold. Deng et al. (2026) discuss conditions under which small drift suggests approximate matching. Vault's finite-sample field uses separate bidirectional normalizations and self-masked negatives, so it is not established to inherit the reference field's exact anti-symmetry or distribution-matching guarantees.

Kernel mean-shift instantiation. The practical field of Eq. equation 1 is a pair of kernel-weighted mean shifts: $\mathbf{V}_p^+$ attracts a sample toward the proximity-weighted mean of the positives $\mathbf{y}^+ \sim p$, $\mathbf{V}_q^-$ toward that of the negatives $\mathbf{y}^- \sim q$, and $\mathbf{V}_{p,q} = \mathbf{V}_p^+ - \mathbf{V}_q^-$ attracts by the data distribution and repels by the generated distribution. The kernel $k(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{x} - \mathbf{y}\|_2/\tau)$ is realized as a softmax over samples with logits $-\|\mathbf{x} - \mathbf{y}\|_2/\tau$, in the spirit of InfoNCE. The softmax normalizer is exactly the expectation $\mathbb{E}[k]$ in the denominator of Eq. equation 1. The reference implementation additionally applies a softmax over the batch of generated samples (a bidirectional normalization that slightly improves performance in practice), and draws the negatives from the generated batch itself, masking each sample's self-match.

Training objective. The fixed-point relation $f_{\hat{\theta}}(\boldsymbol{\epsilon}) = f_{\hat{\theta}}(\boldsymbol{\epsilon}) + \mathbf{V}_{p,q_{\hat{\theta}}}(f_{\hat{\theta}}(\boldsymbol{\epsilon}))$ at equilibrium motivates the regression of Eq. equation 2: the target is the current sample displaced by one drift step and frozen by the stop-gradient. The loss value equals the squared field norm, $\mathbb{E}\|\mathbf{V}\|^2$, but gradients never flow through $\mathbf{V}$ (which depends on the distribution q_θ). The solver minimizes the objective indirectly by moving each sample toward its frozen, displaced self.

Feature-space drifting. The objective can be defined in any feature space: with an encoder ϕ, the positives and negatives are $\phi(\mathbf{y}^+)$ and $\phi(\mathbf{y}^-)$, and the loss regresses $\phi(\mathbf{x})$ toward $\mathrm{sg}(\phi(\mathbf{x}) + \mathbf{V}(\phi(\mathbf{x})))$. The original work uses pre-trained self-supervised encoders and stresses that feature encoding is a training-time operation, absent at inference. Vault instantiates exactly this form with $\phi = \mathrm{VAE}_{\mathrm{enc}}$ (Appendix C motivates this choice of space), and the kernel proximity then measures behavioral similarity between trajectories.

Distribution-level guidance. Drifting also admits a classifier-free-guidance-style behavior at training time. With class (scene) condition c, the negatives are drawn from a mixture $\tilde{q}(\cdot \mid c) = (1-\gamma)\, q_\theta(\cdot \mid c) + \gamma\, p_{\text{data}}(\cdot \mid \varnothing)$ that mixes generated samples with *real* unconditional data. Driving $\tilde{q}(\cdot \mid c)$ toward $p_{\text{data}}(\cdot \mid c)$ then requires the conditional generator to match

$$q_\theta(\cdot \mid c) = \alpha\, p_{\text{data}}(\cdot \mid c) - (\alpha - 1)\, p_{\text{data}}(\cdot \mid \varnothing), \qquad \alpha = \tfrac{1}{1-\gamma} \geq 1, \tag{6}$$

so $\alpha = 1$ is pure conditional matching and $\alpha > 1$ extrapolates away from the unconditional distribution. This is a training-time mechanism. One-step inference is unaffected.

B Instantiating the Drift Field in Vault

This section details how Section 3.2 instantiates the theory above, and collects the derivations omitted from the main text.

B.1 Anchors, Negatives, and Scale Normalization

The conditional anchor set $\mathcal{A} = \{\mathbf{z}_{\text{GT}}\} \cup \{\text{pool positives}\}$ contains one ungated expert latent and any available pool entries that pass the maximum-score gate (Section 3.3). The unconditional branch instead uses the scene-agnostic vocabulary, which is not guaranteed feasible in the current scene. The generated distribution is represented by the N candidates of the current pass, with the other candidates $\mathcal{N}_i = \{\mathbf{z}_j\}_{j \neq i}$ as negatives for candidate i.

Latent distances are uncalibrated: their absolute scale depends on the VAE and their concentration on the latent dimension D. Both sides of every kernel are therefore evaluated on distances divided by the scene scale

$$S_{\text{sc}} = \mathrm{sg}\Big(\max\{\overline{d(\mathbf{z}, \mathcal{A})}/\sqrt{D}, \varepsilon_s\}\Big), \tag{7}$$

where the bar averages candidate-to-anchor distances over all pairs in a scene and $\varepsilon_s > 0$ is a numerical floor. This makes kernel bandwidths relative to the current latent scale. The value of the floor must be reported with the implementation. The $\sqrt{D}$ term changes how bandwidths vary with latent dimension but does not make the learned space scale-free.

B.2 Bidirectional Assignment and Kernel Mean Shift

With anchors as an empirical measure of p, the row-softmax of Eq. equation 3,

$$r_{ij} = \frac{e^{-d(\mathbf{z}_i,\mathbf{a}_j)/(\tau_\ell S_{\text{sc}})}}{\sum_{j'} e^{-d(\mathbf{z}_i,\mathbf{a}_{j'})/(\tau_\ell S_{\text{sc}})}} = \frac{k(\mathbf{z}_i,\mathbf{a}_j)}{\sum_{j'} k(\mathbf{z}_i,\mathbf{a}_{j'})}, \tag{8}$$

would give the empirical kernel mean $\sum_j r_{ij}\mathbf{a}_j$ and displacement $\sum_j r_{ij}\mathbf{a}_j - \mathbf{z}_i$. Vault instead uses w^+_{ij} from Eq. equation 3, which also includes a column normalization. Thus $\mathbf{A}_i = \sum_j w^+_{ij}\mathbf{a}_j$ is a modified weighted mean, not exactly the empirical field in Eq. equation 1. The negative side is modified analogously.

The column-softmax c_{ij} normalizes over candidates for each anchor, and $\sqrt{r_{ij}c_{ij}}$ combines the two directions before row renormalization. Unlike the reference implementation of Deng et al. (2026), Vault normalizes the anchor and negative blocks separately. Each resulting weight vector is nonnegative and sums to one, making $\mathbf{A}_i$ and $\mathbf{R}_i$ weighted means. This changes the reference field and its theoretical properties. Many-to-one assignment permits several candidates to share an anchor. Repulsion creates a separation pressure but does not rule out a collapsed solution.

B.3 Multi-Bandwidth Kernel

Small bandwidths concentrate weights near the closest anchors, while large bandwidths average more broadly. Both remain convex combinations. Vault sums displacements over $\{\tau_\ell\} = \{1, 2, 4\}$, following the multi-temperature idea of Deng et al. (2026). This reduces dependence on a single bandwidth but does not remove sensitivity to their chosen values. Any fixed-point condition for the implemented objective applies to the summed field.

B.4 Adaptive Step Size

The common denominator $\lambda_i = \text{sg}\big(\max\{\sqrt{\overline{\|\mathbf{V}\|_2^2}/D}, \varepsilon\}\big)$ uses the batch RMS field magnitude and a positive numerical floor. For RMS above the floor, substituting the same current field into Eq. equation 5 yields a scalar value of exactly one. Nevertheless, the stop-gradient target gives $\partial\mathcal{L}/\partial\mathbf{z}_i = -2\mathbf{V}_i/(ND\lambda_i)$, treating the target and denominator as fixed. The equation is therefore an update surrogate, not a conventional energy whose scalar value tracks convergence. At a zero field the floor makes the update and loss well-defined and zero.

B.5 Equilibrium Condition

At $\alpha = 1$ the unconditional term cancels. For one bandwidth, a zero-field configuration satisfies $\mathbf{V}_i = \mathbf{A}_i - \mathbf{R}_i = 0$, or

$$\sum_j w^+_{ij}\,\mathbf{a}_j \;=\; \sum_{m\neq i} w^-_{im}\,\mathbf{z}_m \qquad \text{for every candidate } i: \tag{9}$$

This equality constrains the two weighted means, not the location of $\mathbf{z}_i$ or the number of occupied modes. With multiple bandwidths, only the sum of fields needs to vanish. Consequently, coverage and quality must be established by the empirical candidate and oracle analyses rather than inferred from the fixed-point equation.

B.6 The α Family

Eq. equation 5 trains one network over $\alpha_i \sim \mathcal{U}[0, 2]$. At $\alpha = 0$ the field uses the scene-agnostic vocabulary, at $\alpha = 1$ it uses the current scene's expert and pool anchors, and above one it extrapolates the conditional drift direction. The algebra resembles classifier-free guidance applied to field directions, but it is not the distribution-level guidance construction of Eq. equation 6: Vault changes the positive anchor set and leaves the co-generated negatives fixed. Values below one make the deployed value $\alpha = 1$ interior to the training range. The scene-agnostic vocabulary may include trajectories infeasible in the current scene, though it cancels from the inference field at $\alpha = 1$.

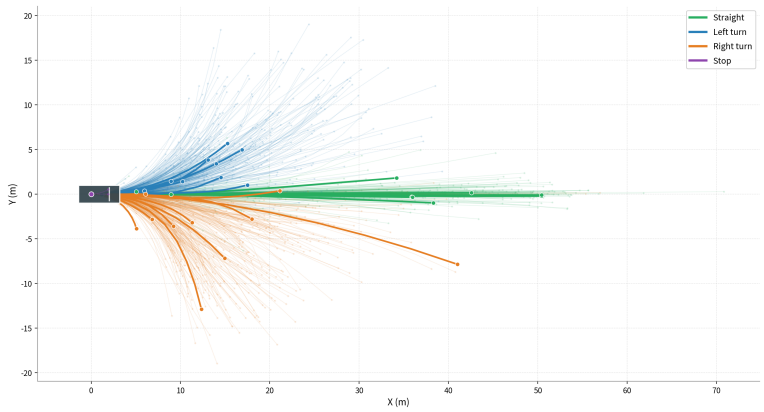


Figure 4: Scene-agnostic vocabulary in bird's-eye view, color-coded by maneuver class (left turn, straight, right turn, stop). Light trajectories are all logged expert samples, and highlighted trajectories are the per-step draws used as unconditional anchors during simulated training, allocated in a 5:5:5:1 class ratio with $K = 32$.

B.7 Scene-Agnostic Vocabulary

The unconditional branch of Eq. equation 5 draws its anchors from a scene-agnostic vocabulary $\mathcal{V}$ built before planner training. All logged expert trajectories of navtrain are grouped by navigation command into four maneuver classes, namely left turn, straight, right turn, and stop, and each class is clustered separately, yielding 512 prototypes for each of the left-turn, straight, and right-turn classes and 32 for the stop class. At each training iteration, $K = 32$ prototypes are drawn from $\mathcal{V}$ as the unconditional anchor set, allocated across the four classes in a 5:5:5:1 ratio. The vocabulary is frozen after construction, the prototypes are shared across scenes, and the branch has zero weight at the deployed $\alpha = 1$. Figure 4 illustrates the construction.

C Drift Space Analysis

The training-time drift is evaluated in the space of a trajectory VAE fitted with reconstruction and auxiliary maneuver-classification losses. We compare this space with physical waypoints and a PCA projection. The decoder still emits physical trajectories through its PCA basis at inference.

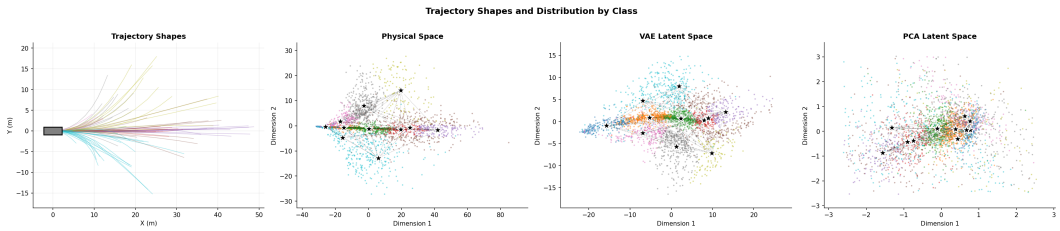


Figure 5: Trajectory shapes and 2D projections in three candidate drift spaces. The plotted sample suggests greater maneuver separation in the VAE representation, quantified by Table 3.

For labeled maneuver classes such as left turn, right turn, and straight, Table 3 reports intra-class and inter-class distances and silhouette coefficients. Distance magnitudes depend on each space's units and scaling. The silhouette coefficient is more directly comparable here.

The VAE space has the highest silhouette in this analysis (0.152, versus 0.025 in physical space and -0.458 in PCA space), supporting its use for the drift kernel on these samples. The separation is modest and does not prove that kernel averages preserve maneuver semantics or trajectory feasibility. PCA remains the output basis, while the VAE is a training-time feature map. A further check would

Table 3: Maneuver-class distance statistics for three candidate drift spaces. Distances have different units across spaces. Silhouette is dimensionless, and higher indicates better separation on these samples.

Space	Intra-Class	Inter-Class	Inter/Intra	Silhouette
Physical Space	7.98	28.12	3.52	0.025
VAE Latent Space	4.45	12.99	2.92	0.152
PCA Space	2.04	2.18	1.07	-0.458

measure whether VAE reconstruction and PCA projection preserve rare turns and out-of-distribution expert trajectories.

D Vault Scorer Training Objective

The scorer is trained with a pairwise hinge ranking loss on the overall score,

$$\mathcal{L}_{\text{rank}} = \frac{1}{\max\{1, |\mathcal{P}|\}} \sum_{(i,j)\in\mathcal{P}} \max\big(0,\, \delta - (S_i - S_j)\big), \tag{10}$$

where $\mathcal{P} = \{(i, j) : y_i - y_j > \delta\}$ is the ordered set of pairs whose label difference exceeds the margin. The denominator also defines the loss for an empty pair set. With $\mathcal{M}$ denoting the sub-metrics, a regression term is

$$\mathcal{L}_{\text{mse}} = \frac{1}{N}\sum_{i=1}^{N}(S_i - y_i)^2 + \frac{1}{N|\mathcal{M}|}\sum_{i=1}^{N}\sum_{m\in\mathcal{M}}\big(S_i^{(m)} - y_i^{(m)}\big)^2, \tag{11}$$

and the total objective is $\mathcal{L}_{\text{scorer}} = \lambda_{\text{rank}}\mathcal{L}_{\text{rank}} + \lambda_{\text{mse}}\mathcal{L}_{\text{mse}}$ with $\lambda_{\text{rank}} = 0.5$ and $\lambda_{\text{mse}} = 1$. The pairwise term encourages correct relative ordering, while the regression term fits the overall score and auxiliary sub-metrics. Whether these terms improve top-one selection requires a separate scorer ablation.

E NAVSIM Metric Definitions

The v1 per-scenario raw PDM score combines these components before benchmark aggregation:

$$s_i = \mathrm{NC}_i\,\mathrm{DAC}_i\,\frac{5\,(\mathrm{EP}_i + \mathrm{TTC}_i) + 2\,\mathrm{C}_i}{12}, \tag{12}$$

Here NC is no at-fault collision, DAC is drivable-area compliance, EP is ego progress, TTC is time-to-collision, and C is comfort. NC and DAC are multiplicative gates, and the benchmark aggregates s_i into the reported PDMS. The v2 per-scenario raw extended score adds driving-direction compliance (DDC), traffic-light compliance (TLC), lane keeping (LK), history comfort (HC), and extended comfort (EC):

$$q_i = \widetilde{\mathrm{NC}}_i\,\widetilde{\mathrm{DAC}}_i\,\widetilde{\mathrm{DDC}}_i\,\widetilde{\mathrm{TLC}}_i\,\frac{5\,(\widetilde{\mathrm{EP}}_i + \widetilde{\mathrm{TTC}}_i) + 2\,(\widetilde{\mathrm{LK}}_i + \widetilde{\mathrm{HC}}_i + \mathrm{EC}_i)}{16}, \tag{13}$$

The tilde denotes the corrected official human-reference filter: for a filtered metric, a violation by the human reference suppresses the corresponding agent penalty on that scenario. It affects multiple submetrics, not just ego progress. EC is computed separately. The official evaluator performs the final split-specific aggregation. EPDMS* in Table 2 uses the original evaluator. EPDMS uses the corrected implementation. We compare methods only within an evaluator column.

F Reported Implementation Details

The reported planner uses four camera views over a sixteen-frame history at 384×672 resolution, $N = 64$ candidates during training and $N = 1024$ at headline inference, up to $M = 32$ conditional

anchors and $K = 32$ unconditional vocabulary prototypes per training iteration, three kernel bandwidths $\{1, 2, 4\}$, and training-time $\alpha \sim \mathcal{U}[0, 2]$ with inference at $\alpha = 1$. Training uses AdamW with a learning rate of 10^{-4} for 30 epochs on eight RTX 4090 GPUs, with a global batch size of 128 realized by gradient accumulation. All reported runs use random seed 1234. The Vault Decoder has 8 layers with feature width $D_{\text{dec}} = 256$. The trajectory VAE has latent dimension $D = 8$, and the frozen PCA output basis has 8 components. The pool is refreshed once per epoch, admitting every candidate that attains the maximum score on its training scene. The VAE encoder and PCA basis are frozen during planner training. The visual encoder remains trainable. The scorer predicts overall PDMS and component scores. End-to-end inference at $N = 1024$ takes 91.4 ms on an RTX 4090, with component timings of 76.3 ms for the visual encoder, 5.4 ms for the decoder, and approximately 9 ms for the scorer.