HYDRA: A NAVIGATION WORLD ACTION MODEL WITH DISCRETE LATENT PLANNING AND CONTINUOUS FLOW-MATCHING EXECUTION

Anonymous authors
Paper under double-blind review

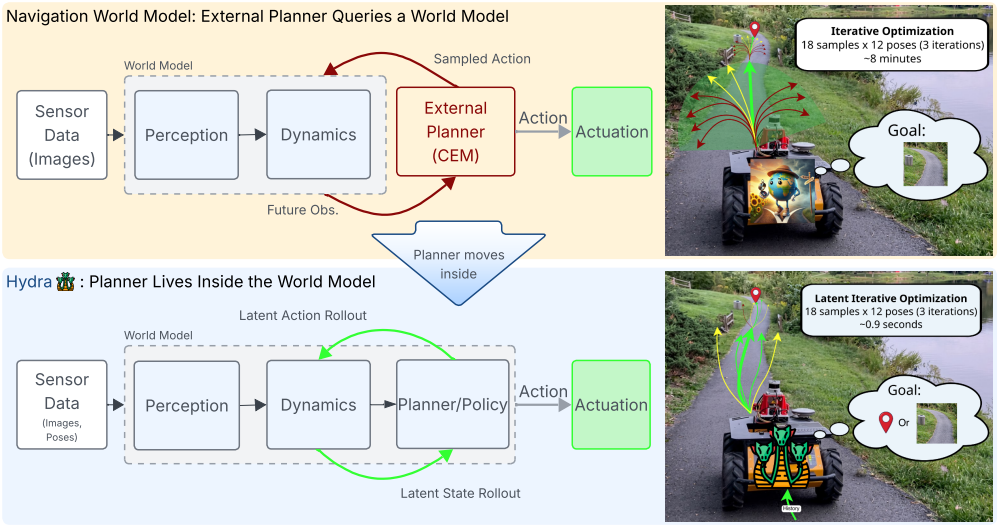


Figure 1: **Moving the planner into the model's own latent space. Top:** NWM uses an external planner to sample unconstrained trajectories, decoding them to pixels for evaluation (red), wasting compute on infeasible candidates. **Bottom:** Hydra plans directly within its discrete latent manifold, then executes via continuous Flow Matching, achieving $> 500\times$ faster planning on a robot.

ABSTRACT

World models let robots imagine possible futures, but exploiting this capability for real-time planning is bottlenecked by a *representation misalignment*: generative models and planners operate on decoupled manifolds, requiring computationally expensive decoding of every candidate back to the high-dimensional observation space for evaluation. In this paper, we present Hydra, a discrete World Action Model that tackles this by establishing a unified latent manifold over visual states, physical poses, and control actions. By compressing this manifold through modality-specific Vector-Quantized bottlenecks, Hydra yields discrete vocabularies of kinodynamic intents and visual states. This enables *Discrete Latent Planning* (DLP), where candidates are sampled directly from the shared manifold and ranked by a Kinematic-Perceptual Cost entirely within the discrete latent space. To bridge discrete planning with the continuous commands required for physical actuation, Hydra pairs DLP with conditional Flow Matching to map selected intents to smooth execution trajectories. Evaluated on two physical robotic platforms, Hydra outperforms state-of-the-art navigation world models in goal-directed planning, while matching or exceeding the closed-loop execution capabilities of leading reactive navigation policies[1].

[1]Model weights and deployment code will be publicly available after the review process.

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.

1 INTRODUCTION

Visual foundation policies (Shah et al., 2023a;b; Sridhar et al., 2023) successfully map high-dimensional observations to motor commands for short-horizon, image-goal navigation, where the goal is visually reachable. However, their reactive nature limits their ability to plan ahead: lacking any mechanism to reason beyond immediate visual cues, making them prone to wandering off when the goal is not observable. Furthermore, the lack of an explicit cost formulation restricts their ability to adapt without policy retraining.

Recent approaches use World Models (WMs) (Ha & Schmidhuber, 2018; Bar et al., 2025) to predict future states conditioned on candidate actions from external planners like CEM (Rubinstein, 1997) or MPPI (Williams et al., 2015), giving the agent the foresight reactive policies lack. However, this introduces a fundamental representation misalignment: planners search an unbounded, continuous space decoupled from the model's latent manifold, requiring computationally expensive decoding to raw observations to evaluate candidates.

To bridge this gap, we introduce Hydra, a World Action Model (WAM) that embeds the sampling-based planner and candidate trajectory evaluator entirely within a unified latent manifold (Fig. 1). Because sampling now draws directly from this joint manifold, candidate actions are informed by the model's own understanding of the observation rather than proposed blindly; because the manifold is bounded, search remains manageable across longer horizons; and because evaluation happens within this shared space, candidates are scored without ever decoding back to pixels.

To achieve this, Hydra feeds its future predictions through modality-specific Vector Quantized (VQ) (Van Den Oord et al., 2017) codebooks, compressing the unbounded, continuous action space into a discrete manifold of *intents* – macro-level kinodynamic primitives, such as a sweeping left turn, drawn from a small vocabulary learned from the training data. This substitution changes what planning actually is: rather than optimizing over a continuous space in which most candidate paths are physically implausible, Hydra searches over a finite set of behaviors already known to be plausible, with the code retrieved at each step being the one whose embedding lies closest to Hydra's embedding of the observation history, rather than an arbitrary vocabulary entry.

We term this search *Discrete Latent Planning* (DLP). As Fig. 1 illustrates, this changes not only how many candidates are considered but what kind: NWM's continuous sampler (red) spreads trajectories indiscriminately, evaluating every one to discover which are bad; Hydra's candidates (green) are drawn only from codes the model has already learned to associate with the observed scene, so implausible trajectories are rarely sampled to begin with.

Because DLP operates in the discrete latent space, candidate evaluation is efficient. Hydra utilizes a Kinematic-Perceptual Cost ($\mathcal{J}_{\text{KPC}}$) where Shannon entropy represents predictive uncertainty, and out-of-distribution (OOD) visual quantization error acts as a direct proxy for anomalous states, such as impending collisions, without ever decoding to pixels. The selected discrete intent is subsequently passed to continuous Flow Matching ODE solvers, integrating the macro-level intent into the continuous motor commands required for smooth physical execution. Ultimately, Hydra demonstrates capability as both a planner and a reactive path follower, matching the execution precision of reactive navigation policies while outperforming baseline decoupled planner and World Models.

In summary, our main contributions are: 1) **Hydra:** A multimodal World Action Model that unifies visual, kinodynamic, and action modalities into a single latent space, moving the planner, the search space, and cost evaluation into the model's unified latent manifold rather than leaving it external and decoupled. 2) **Discrete Latent Planning:** An iterative latent search, guided by a multi-objective Kinematic-Perceptual Cost, that avoids costly decoding to observations with search and safety evaluation within a discrete manifold. 3) **Real-World Deployment:** Hydra functions as both a planner and a reactive path follower, outperforming NWM (Bar et al., 2025) in planning while matching the execution precision of dedicated reactive navigation policies.

2 RELATED WORK

To show the contributions of Hydra, we review prior literature across three dimensions: *Reactive Navigation Policies*, which lack the generative foresight required for planning; *Generative World*

Models, which predict but often lack explicit Euclidean grounding; *Planning with a Model*, where decoupled sampling imposes computational bottlenecks that Hydra's discrete latent planning avoids.

Reactive Navigation Policies: Driven by large-scale heterogeneous datasets, recent efforts have produced capable end-to-end navigation policies (Shah et al., 2023a;b; Sridhar et al., 2023; Liu et al., 2025), including diffusion-based approaches that model multimodal action distributions (Chi et al., 2025; Nguyen et al., 2025; Zheng et al., 2025b; Pan et al., 2026). These architectures are fundamentally reactive, compressing navigation into a direct observation-to-action mapping without an internal model of future consequences. Because they are open-loop with respect to the future environment, they cannot evaluate or prune candidate trajectories before execution, making them myopic and incapable of planning.

Generative World Models: To endow robots with foresight, WMs (Ha & Schmidhuber, 2018; Wu et al., 2023; Janner et al., 2022; Yu et al., 2023; NVIDIA et al., 2025; Park et al., 2026; Zhu et al., 2025b; Maes et al., 2026; Kim et al., 2026; NVIDIA, 2026) simulate the consequences of actions, evolving from early predict-then-act pipelines Du et al. (2023); Wen et al. (2024); Bharadhwaj et al. (2025) toward unified diffusion (Sohl-Dickstein et al., 2015; Rombach et al., 2022; Hoogeboom et al., 2022) or Flow Matching (Liu et al., 2022; Lipman et al., 2023; Xing et al., 2025) architectures. Visual-action models (Hu et al., 2023; Russell et al., 2025; Bar et al., 2025; Brooks et al., 2024) and large-scale unified backbones (Li et al., 2025; Zhu et al., 2025a; Ali et al., 2025; Octo Model Team et al., 2024; Mereu et al., 2025; World Labs, 2025; Team et al., 2026a;b) provide tighter coupling but typically at billions of parameters for general-purpose manipulation; DreamZero (Ye et al., 2026), for instance, uses a 14B-parameter WAM as a zero-shot policy on physical manipulators, performing no search since rollout-based planning at this scale is computationally prohibitive. NavWM (Mei et al., 2026) targets navigation with a Mamba backbone, but discretizes only candidate goal locations via a K-means anchor codebook; so evaluation still requires generated visual rollouts rather than a decode-free latent score. Hydra targets navigation instead, at 143M parameters, sized for real-time search.

Planning with a Model: Sampling-based planning approaches such as MPPI or CEM (Zhou et al., 2024; Bar et al., 2025; Maes et al., 2026) use World Models as the dynamics forward model; for navigation, NWM (via CEM) and VertiFormer (Nazeri et al., 2025) (via MPPI) are our closest baselines. Both search an unconstrained continuous action space, but the consequences differ by architecture: lacking a native action head, NWM must decode each candidate and evaluate it with a perceptual metric such as LPIPS (Zhang et al., 2018), which is prone to texture-matching illusions rather than measuring true spatial distance; VertiFormer instead retains a direct geometric cost from its action head, but its unbounded search space still admits implausible, collision-prone candidates absent a structural prior. Latent-space planning (Fang et al., 2023; Zheng et al., 2025a; Su et al., 2026; Zhou et al., 2024; Sobal et al., 2025; Assran et al., 2025; Sun et al., 2026) avoids pixel rendering, but JEPA-style models (Assran et al., 2023; Bardes et al., 2023; Zhu et al., 2025b; Assran et al., 2025; Balestriero & LeCun, 2025) still evaluate continuous action manifolds, inheriting the same sample inefficiency. Discretizing the latent space is not itself new; DreamerV3 (Hafner et al., 2023) and IRIS (Micheli et al., 2023) do so, but neither couples discretization to a candidate-ranking cost function nor targets physical, real-time robot planning. WorldPlanner (Khorrambakht et al., 2025) discretizes the Monte Carlo Tree Search rather than the underlying representation, so actions and states remain continuous and pixel-valued, requiring decode-then-evaluate and taking five minutes on average to plan. LeFlow (Huang et al., 2026), applied to toy problems, amortizes search by learning a rectified-flow prior that generates a goal-conditioned latent path in one forward pass; built on a JEPA backbone with no visual decoder, it avoids decoding architecturally, but its representation remains continuous and unconstrained. Hydra discretizes the representation itself: kinodynamic intents and visual states are drawn from finite, learned vocabularies, so search and cost evaluation happen entirely within this discrete space, on a physical robot rather than in simulation.

3 HYDRA

This section presents Hydra in four parts. We first formalize autonomous navigation through the lens of three paradigms (see Fig. 2), situating Hydra's unified planning relative to reactive policies and decoupled world models (Sec. 3.1). We then detail Hydra's architecture, specifically the discrete codebook (Sec. 3.2). Next, we explain how Hydra conducts Discrete Latent Planning within these

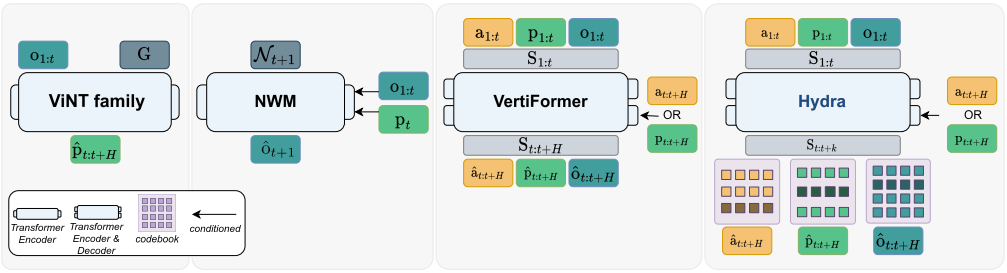


Figure 2: Comparison of Hydra architecture with Visual Navigation Transformer (ViNT), Navigation World Model (NWM), and VertiFormer in terms of input modalities, conditioning, and the model output. Hydra builds on top of VertiFormer and discretizes the output into codebooks to enable the model to search the discrete latent space for a plausible path.

codebooks, searching and ranking trajectories without pixel-level decoding (Sec. 3.3). Finally, we describe Hydra's local path following strategies when guided by an external global planner (Sec. 3.4).

3.1 Problem Formulation: Navigation Paradigms

Autonomous navigation requires predicting an action sequence to achieve a goal in a partially observable environment. At timestep t, the model receives a multimodal observation $O_t = \{I_t, p_t, a_{t-1}\}$, comprising ego-centric vision $I_t \in \mathbb{R}^{3\times H\times W}$, odometric pose $p_t = (x_t, y_t, \theta_t)$, and the prior motor command $a_{t-1} = (v, \omega)$.

The agent maintains a context history $\mathcal{H}_t = \{O_{t-K+1}, \ldots, O_t\}$ over a window K. Based on how models map this history to actions, we categorize navigation models into three formulations. While we detail *Reactive Policies* and *Decoupled Planning* in App. A, the third paradigm, *Unified Planning*, forms the basis of Hydra and is described below.

World Action Model Formulation (Unified Planning):

Hydra formulates planning so that the planner and the model share the same embedding: visual states, poses, and actions are all sampled and evaluated within a single latent manifold, as opposed to a decoupled model where the planner sends actions to and receives images from the model.

Given the observation history $\mathcal{H}_t$ and a sampled sequence of future actions $a_{t:t+H-1}$ or poses $p_{t+1:t+H}$, the model P_{WA} generates the expected future observations:

$$(\hat{I}_{t+1:t+H}, \hat{y}) \sim P_{WA}(\cdot \mid \mathcal{H}_t, x) \tag{1}$$

where

$$(\hat{y}, x) = \begin{cases} (\hat{p}_{t+1:t+H}, a_{t:t+H-1}) & \text{if forward kinodynamics,} \\ (\hat{a}_{t:t+H-1}, p_{t+1:t+H}) & \text{if inverse kinodynamics.} \end{cases}$$

3.2 Hydra

In this section, we discuss the three primary mechanisms of Hydra: unified early-fusion encoding, discrete decoding, and multimodal flow matching. Figure 2 shows Hydra's architecture compared with previous approaches. In the Appendix, Table 4 compares Hydra's modalities with existing approaches.

Unified Multimodal Encoding via Early Fusion. Visual latents $z_i \in \mathbb{R}^{D_z}$ are extracted via a frozen VAE (Blattmann et al., 2023), and low-dimensional actions and poses undergo Fourier Feature Mapping (Tancik et al., 2020; Zhou et al., 2023) to capture sharp control transitions. To avoid computationally heavy cross-attention between vision and kinodynamics, Hydra adopts an early-fusion unified encoding strategy akin to VertiFormer (Nazeri et al., 2025). More details are provided in Appendix D.

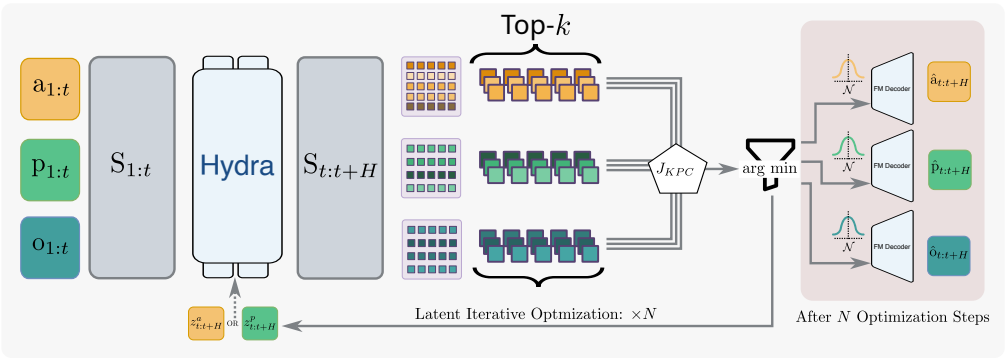


Figure 3: Initial intended future pose/action tokens are generated unconditionally, then Gaussian noise is added to become the exploratory seed 'intents'. Then these future intents go through VQ codebooks, and we sample the top-k codes. After calculating the KPC cost, the lowest cost is chosen to seed the next iteration of searching. The best trajectory is decoded using flow matching heads to continuous robot poses and actions at the end for robot execution.

We concatenate this unified history stream with H learnable context tokens, where H is the forecasting horizon. The Transformer Encoder processes this joint sequence and outputs a fixed-size set of context tokens, $\text{ctx}_{t+1:t+1+H}$. These tokens compress the "where, what, and how" of the robot's history.

The Discrete Bottleneck. The Transformer Decoder cross-attends the context tokens $\text{ctx}_{t+1:t+1+H}$ with the intended future poses ($p_{t+1:t+1+H}$) or actions ($a_{t:t+H}$), utilizing the same missing-modality infilling as the encoder. This yields H "intent" tokens (z_e), which are subsequently fed to modality-specific VQ codebooks (Van Den Oord et al., 2017; Lee et al., 2022; Zargarbashi et al., 2026), one per modality. To balance high-fidelity spatial forecasting with the strict latency constraints of closed-loop control, we employ *asymmetric vocabulary quantization*. We allocate a large codebook ($|\mathcal{V}_i| = 2048$) to the visual generative head, while strictly constraining the kinodynamic intents ($|\mathcal{V}_p| = 64, |\mathcal{V}_a| = 64$). This asymmetry preserves visual representational capacity while explicitly enabling the fast retrieval speeds required for DLP. To stabilize training, we applied deterministic anchoring; see App. C for details.

Multimodal Generative Flow Matching. For each modality $m \in \{\text{img,pose,act}\}$, we define a probability path from a source Gaussian distribution $\psi_0^{(m)} \sim \mathcal{N}(0, I)$ to the ground-truth target $\psi_1^{(m)}$. At any continuous integration time $s \in [0, 1]$, the intermediate noisy state is $\psi_s^{(m)} = (1 - s)\psi_0^{(m)} + s\psi_1^{(m)}$. Hydra employs three flow decoders (Liu et al., 2022), one per modality, denoted $v_\theta^{(m)}$, to predict the target velocity field for images, poses, and actions.

Each discrete token $c^{(m)}$, the index selected from the VQ codebook, is used to retrieve its corresponding embedding $e_c^{(m)}$. This embedding, along with the time step s, conditions the flow decoders via DiT blocks (Peebles & Xie, 2023). The network is optimized end-to-end via the vector field loss, summed over modalities:

$$\mathcal{L}_{flow} = \sum_m \mathbb{E}_{s,\psi_0^{(m)},\psi_1^{(m)}} \left[|v_\theta^{(m)}(\psi_s^{(m)}, s, e_c^{(m)}) - (\psi_1^{(m)} - \psi_0^{(m)})|_2^2 \right].$$

Please refer to Appendix C for more details about the design choices.

3.3 Planning in the Discrete Latent Space

In decoupled planning and world models, candidate actions are proposed without any knowledge of the observed scene, and evaluating them requires decoding back to the raw observation (*e.g.*, pixels) rather than comparing candidates directly in latent space. Hydra closes this gap by coupling the planner to the model itself, introducing *Discrete Latent Planning* (DLP) (equation 1) in place of the decoupled formulation in equation 6. Instead of searching blindly in a continuous space, Hydra operates explicitly over the learned manifold defined by the discrete modality codebooks $\mathcal{V}_i$, $\mathcal{V}_p$ and $\mathcal{V}_a$.

Unbiased Initialization and Gumbel-Max Search. In the absence of an explicit future intent, Hydra initializes its search using an *Unbiased Expert Prior* by querying the model with masked future states. Using this unbiased seed, Hydra generates a diverse batch of candidate paths by adding Gaussian noise. Furthermore, by applying exponentially smoothed Gumbel noise to the discrete codebook logits, Hydra injects path perturbations while preserving kinodynamic smoothness across time steps.

Latent Optimization. Once candidates are retrieved, they must be ranked. Previous visual planning models such as NWM used perceptual similarity such as LPIPS (Zhang et al., 2018), which incur massive computational overhead due to the need to decode the latents to pixels for computation. To avoid this, Hydra evaluates candidates within the discrete manifold using a multi-objective cost function, defined for a candidate sequence τ as:

$$\mathcal{J}_{\text{KPC}}(\tau) = \underbrace{\mathcal{C}_{\text{geo}}}_{\text{Geometric Goal}} + \sum_{t=1}^{T} \gamma^t \left(\underbrace{\lambda_{\text{img}} \mathcal{C}_{\text{img}}^{(t)}}_{\text{Visual Goal}} + \underbrace{\lambda_{\text{prior}} \mathcal{C}_{\text{prior}}^{(t)}}_{\text{Kinodynamic Prior}} + \underbrace{\lambda_{\text{ent}} \mathcal{C}_{\text{ent}}^{(t)} + \lambda_{\text{vq}} \mathcal{C}_{\text{vq}}^{(t)}}_{\text{Perceptual Safety}} \right) \tag{2}$$

where $\gamma \in (0, 1)$ is a temporal discount factor that decays the weight of distant future states to account for compounding generative errors.

- **Geometric and Semantic Tracking ($\mathcal{C}_{\text{geo}}, \mathcal{C}_{\text{img}}$):** both are evaluated as L_2 deviations from the goal, $\mathcal{C}_{\text{geo}}$ from the local goal coordinates and $\mathcal{C}_{\text{img}}$ from the visual goal latent embedding. A direct path minimizes this cost, but requires regularization to survive occlusions.
- **Unbiased Expert Prior ($\mathcal{C}_{\text{prior}}$):** penalizes deviations from collision-free demonstrations via the Negative Log-Likelihood (NLL) of sampled tokens.
- **Visual Predictive Entropy ($\mathcal{C}_{\text{ent}}$):** computes the Shannon entropy of the predicted visual token distribution, penalizing trajectories that enter ambiguous or unobservable regions.
- **Manifold Rejection ($\mathcal{C}_{\text{vq}}$):** computes the quantization error between the continuous sampled intent and the visual codebook $\mathcal{V}_i$. OOD states (*e.g.*, driving into an occlusion) fall off the learned manifold of safe navigation, yielding high quantization error and allowing rejection without invoking the visual decoder.

Note that all KPC terms are calculated in the latent space except $\mathcal{C}_{\text{geo}}$, which is decoded using the deterministic head to compute the Euclidean distance to the goal, a negligible computational cost. By evaluating $\mathcal{J}_{\text{KPC}}(\tau)$ on the intent tokens, iterative refinement runs only the transformer decoder. Flow matching is deferred, called only once to decode the chosen trajectory ($N = 10$ NFEs for kinodynamic; $N = 50$ for visual). This latent formulation drives a $> 500\times$ reduction in planning time relative to NWM (Table 2). Finally, the discrete tokens of this trajectory are decoded into local spatial deltas and fused via Action Chunking (Zhao et al., 2023) for smooth closed-loop execution.

3.4 Path Following via Global Guidance

While DLP enables standalone planning, long-horizon navigation often requires tracking global subgoals g provided by a planner. Hydra acts as a reactive policy that minimizes deviation from g within obstacle-free regions. We propose two strategies for path following:

Classifier-Free Guidance in Kinodynamic Space. We adapt CFG (Ho & Salimans, 2022) to blend structural safety with goal-directed tracking. We compute an unconditional prior $z_\emptyset$ and a geometric intent z_g, interpolating them:

$$\tilde{z}_g = z_\emptyset + \omega \cdot (z_g - z_\emptyset), \tag{3}$$

with guidance scale ω. However, high ω risks *manifold deviation*, pushing $\tilde{z}_g$ outside the discrete VQ codebook bounds, causing the decoder to hallucinate physically impossible dynamics (App. F.2).

Sub-Goal Sampling. Alternatively, we generate candidate trajectories by perturbing the continuous intent token with Gaussian noise. We decode these into a candidate set and select the trajectory

minimizing the tracking cost:

$$\mathcal{J}_{\text{track}}(\tau) = \|\tau_T - g\|_2 + \frac{1}{2T}\sum_{t=1}^{T}\|\tau_t - g\|_2, \tag{4}$$

where T is the trajectory length. Because candidates are decoded strictly from the expert prior vocabulary, they inherently satisfy learned dynamic constraints and local obstacle avoidance (App. F.3).

4 Experiments

The empirical evaluation of Hydra is designed to answer three core questions: (1) **Generative Grounding:** does Hydra's parameter-efficient, early-fusion architecture maintain competitive visual predictive fidelity against continuous baselines like NWM over long horizons? (2) **Real-Time Planning Efficiency:** does coupling the planner to the model's latent manifold and evaluating candidates without decoding to pixels, bypass the computational bottlenecks of decoupled sampling and enable real-time deployment? (3) **Path-Following Accuracy:** as a local path follower, does Hydra match reactive baselines (*e.g.*, GNM, ViNT, NoMaD) without sacrificing observation-to-action behavior? Qualitative results (App. B.6), pose codebook analysis (App. B.5), and ablations validating the KPC terms (App. B.4 and B.7) are provided in the Appendix.

4.1 Generative Grounding: Long-Horizon Temporal Consistency

To evaluate whether Hydra's early fusion and discrete bottleneck preserve temporal grounding and video generation quality, following NWM, we autoregressively generate 100 video clips of 16 seconds (at 4 FPS) using the same held-out subset of the RECON dataset (Shah et al., 2021). We benchmark against the base variant of NWM (NWM/B, 200M parameters DiT) and VertiFormer (a deterministic baseline), all evaluated under *teacher-forcing* (ground-truth actions at each step) for a fair comparison of the visual generation.

As shown in Table 1, Hydra slightly outperforms NWM in Fréchet Video Distance (FVD), which measures temporal coherence and action adherence. However, both show high variance, depending on whether the robot moves straight. NWM suffers from compounding errors and visibly decouples from the commanded actions after approximately 8 seconds (Appendix B.3), while Hydra's multimodal manifold enforces temporal grounding across the full 16-second horizon. We report single-step predictive fidelity (DreamSim, LPIPS, PSNR) in Appendix B.1.

Table 1: Comparison of Video Synthesis Quality. 100 clips of 16-seconds generated at 4 FPS.

Model	FVD ↓	Time To Generate
NWM/B	22.10 ± 10.71	3.5h
VertiFormer	81.03 ± 12.50	2m
Hydra	$\mathbf{19.15} \pm 5.11$	4m

4.2 Planning: Real-Time Feasibility via Discrete Search

While recent literature benchmarks generative planners using static offline datasets, we argue that this methodology is fundamentally misaligned with the objectives of planning. Offline dataset evaluations penalize any deviation from the human demonstration. If a planner discovers an alternative, physically valid, and collision-free trajectory, static evaluations incorrectly compute a high geometric penalty. True exploratory planning and obstacle avoidance should therefore be evaluated through closed-loop physical deployment. We explain the physical evaluation protocols, including robot type, scenario initialization, and hardware configurations for both planning and local path following in App. G.

Baselines. We benchmark Hydra against VertiFormer (Nazeri et al., 2025), NWM/B, and Hydra-NoVQ, a variant of Hydra that retains Flow Matching heads but removes VQ discretization, isolating the contribution of discrete search from Flow Matching. VertiFormer shares Hydra's early-fusion design but is deterministic, with neither VQ discretization nor Flow Matching, relying on MPPI sampling. All models use the SD's VAE (Blattmann et al., 2023) with 80M frozen parameters. Without a discrete latent representation, neither baseline can leverage our $\mathcal{C}_{\text{prior}}$, $\mathcal{C}_{\text{ent}}$, and $\mathcal{C}_{\text{vq}}$ cost terms. Computing $\mathcal{C}_{\text{prior}}$ and $\mathcal{C}_{\text{ent}}$ requires a normalized probability distribution; over a finite codebook,

this normalization is a tractable sum, whereas over a continuous action space it requires integrating a density over an unbounded domain, which is intractable. $\mathcal{C}_{\text{vq}}$ is unavailable because it measures distance to a learned codebook entry, and no such codebook exists in a continuous representation. Therefore, baselines are restricted to $\mathcal{C}_{\text{geo}}$ and $\mathcal{C}_{\text{img}}$ cost terms.

Table 2: Planning Performance. While VertiFormer with MPPI runs quickly, it suffers from sample inefficiency and is blind to obstacles (0% SR), as is the continuous variant, Hydra-NoVQ. Hydra's DLP achieves high success rates while maintaining real-time execution speeds compared to NWM.

Model	Params	Search Space	Iterations	Samples	Len.	Plan Time ↓	Unobs. (SR) ↑	Obs. (SR) ↑	Corner Turn (SR)↑
NWM/B	200M	Continuous (CEM)	3	18	12	$>$ 500.0s	0	0	0
VertiFormer	27.11M	Continuous (MPPI)	1	18	12	$\sim$ **0.7s**	40	0	0
Hydra-NoVQ	141.60M	Continuous (MPPI)	1	18	12	$\sim$ 1.2s	20	10	10
Hydra	143.29M	Discrete (DLP)	3	18	12	$\sim$ 0.9s	**100**	**80**	**80**

The Failure of External Planners. As shown in Table 2, NWM suffered catastrophic latency under identical hyperparameters (3 iterations, 18 samples, length 12). Due to the lack of a native action head, NWM's one CEM search took over 500 seconds, as it relied on a compute-intensive image goal comparison, which requires each candidate to be decoded to pixels.

VertiFormer and Hydra-NoVQ instead use a direct geometric cost $\mathcal{C}_{\text{geo}}$ from the predicted pose, letting them search orders of magnitude faster ($\sim$0.7, $\sim$1.2s) despite their unbounded space, but both suffer *sample inefficiency*: without a structural prior bounding the search space, they generate collision trajectories, yielding a 0% Success Rate across obstacle scenarios. Because Hydra-NoVQ retains Hydra's Flow Matching heads and differs only in the absence of VQ discretization, this isolates the source of Hydra's obstacle-avoidance capability as the discretization of the search space and the $\mathcal{C}_{\text{ent}}$, $\mathcal{C}_{\text{vq}}$, and $\mathcal{C}_{\text{prior}}$ terms it enables.

Real-Time DLP. Although our code is not optimized to leverage quantization or kernel fusion on GPU, by restricting the search space to a discrete manifold of kinodynamic primitives directly in latent space without decoding, Hydra plans in $\sim$0.9 seconds, a $\sim$500$\times$ speedup over NWM, while achieving a 10/10 success rate in unobstructed environments and 8/10 in both occluded and blind-corner scenarios.

4.3 Local Path Following: Hydra vs. Reactive Policies

We next evaluate Hydra as a *local path follower* that tracks waypoints from an external guidance signal (Sec. 3.4), and compare it against reactive baselines and Hydra operating in planning mode (Sec. 4.2). Table 3 reports Success Rate, Traversal Time, and Interventions on Spot across three scenarios – Obstructed Goal, Long Distance, and Indoor Corridor – for three reactive baselines (GNM, ViNT, and NoMaD), VertiFormer, Hydra's two local path following variants (CFG and Sampling), and Hydra's planner.

Hydra matches reactive baselines despite sparser guidance. GNM, ViNT, and NoMaD have the advantage of accessing a topological map providing a dense stream of intermediate goal images at 2Hz. Conversely, Hydra's variants receive only sparse, widely spaced waypoints; despite this, Hydra matches or exceeds the reactive baselines' success rate and intervention counts across all scenarios tested.

CFG and Sampling trade off differently across horizons. CFG achieves the lowest traversal time and zero interventions on Indoor Corridor, but degrades sharply on Long Distance (138 $\pm$ 7.7s, 2.4 $\pm$ 0.9 interventions): because $\tilde{z}_g$ is computed via continuous interpolation between z_g and $z_\emptyset$ (Sec. 3.4), small per-step deviations compound over long horizons, eventually pushing the latent outside the valid VQ manifold and triggering hallucinated dynamics. Sampling avoids this by restricting sampling candidates to the pose codebook, making it a more reliable choice for long-horizon following at some cost in traversal time.

VertiFormer remains a strong baseline, achieving the fastest traversal time on Obstructed Goal (21.0 $\pm$ 1.05s) and matching the zero intervention count on Indoor Corridor, consistent with a deterministic regression head reaching a single hypothesis directly without search or multi-step rollout, trading collision-avoidance and goal-flexibility (Table 8) for speed. Offline evaluations detailed in App. B.2 exhibit a similar pattern where deterministic baselines edge out generative methods.

Table 3: Local Planning Evaluation across 10 Physical Trials per Scenario. Hydra drastically reduces physical interventions by leveraging its predictive foresight to move around topological traps that ensnare reactive baselines. GNM, ViNT, and NoMaD are conditioned on dense goal-image sequences at 2Hz; Hydra's local-following variants (CFG, Sampling) are conditioned on sparse, more widely spaced waypoints.

		Dense Goal Images (2Hz)			Sparse Waypoints (turns only)			
Scenario	**Metric**	**GNM**	**ViNT**	**NoMaD**	**Hydra (CFG)**	**Hydra (Sampling)**	**VertiFormer**	**Hydra (Planning)**
Obstructed Goal	Success Rate (%) ↑	90	80	**100**	90	50	**100**	90
	Traversal Time (s) ↓	36.4 ± 7.9	34.4 ± 6.8	37.4 ± 7.7	45.0 ± 8.0	48.4 ± 18.5	**21.0** ± 1.05	37.2 ± 21.0
	Interventions ↓	1.1 ± 0.3	1.2 ± 0.6	1.1 ± 0.3	1.0 ± 0.7	**0.8** ± 0.8	1.0 ± 0.5	1.2 ± 0.6
Long Distance	Success Rate (%) ↑	0	0	0	**100**	**100**	**100**	40
	Traversal Time (s) ↓	-	-	-	138 ± 7.7	107 ± 8.6	109 ± 7.6	119 ± 3.8
	Interventions ↓	-	-	-	2.4 ± 0.9	1.2 ± 0.8	1.0 ± 1.0	3.4 ± 0.6
Indoor Corridor	Success Rate (%) ↑	40	90	90	**100**	**100**	**100**	**100**
	Traversal Time (s) ↓	59.0 ± 4.6	48.1 ± 3.3	65.0 ± 21.2	**24.4** ± **2.01**	28.2 ± 0.79	26.3 ± 1.6	33.9 ± 2.0
	Interventions ↓	**0.0** ± 0.0	**0.0** ± 0.0	0.2 ± 0.4	**0.0** ± 0.0	**0.0** ± 0.0	**0.0** ± 0.0	0.1 ± 0.3

Planning vs. local path following. Hydra's planning mode matches or exceeds CFG and Sampling's success rate on Obstructed Goal (90%) and Indoor Corridor (100%), at the cost of higher traversal time and variance from searching at every step. On long distance, however, planning succeeds in only 40% of trials versus 100% for CFG and Sampling: the added per-step search latency is enough to make the robot react late at the scenario's single corridor fork, an unrecoverable failure mode given the trial's time limit even though the same latency is harmless in the shorter scenarios. This is a latency-budget mismatch rather than a failure of the discrete search itself; see App. B.8 for details.

5 Limitations and Future Work

While Hydra demonstrates advancements in latent planning, empirical deployment reveals challenges in DLP. We suggest reading App. B.8 for a further discussion of the limitations.

Manifold Rejection Confuses Visual Complexity with Genuine Risk. Manifold Rejection ($\mathcal{C}_{vq}$) is a useful but imperfect safety signal: it cannot fully distinguish a genuinely dangerous occlusion from a visually complex but entirely safe scene. As shown empirically in Fig. 7, dense, high-frequency textures such as foliage and grass produce quantization error nearly as large as a true physical obstacle, since both are difficult for the visual codebook to reconstruct precisely, one because it is genuinely unfamiliar and dangerous, the other simply because it is visually intricate. We refer the readers to App. B.7 for more details.

Representational Capacity Trades Expressivity for Latency. Hydra's zero-shot generalization is intrinsically tied to the vocabulary size of its VQ codebooks. A larger codebook would capture a wider distribution of visual and kinodynamic behaviors, improving generalization to novel environments, but at a direct computational cost: retrieving the nearest codebook entry during search requires a distance computation whose cost scales with vocabulary size, contradicting the same real-time latency budget that motivated discretization in the first place (Sec. 4.2). Hydra therefore faces a structural trade-off between behavioral expressivity and search latency that a fixed-size codebook cannot resolve on its own; adaptive or hierarchical vocabularies are a natural direction for future work.

6 Conclusion

We presented Hydra, a World Action Model that closes the representation misalignment between generative world models and their planners by moving the planner inside the model's own discrete latent manifold. Across two physical robots, this yields a $\sim$500$\times$ reduction in planning latency over NWM while achieving higher success rates in obstacle-rich environments than every baseline we tested, including Hydra-NoVQ, isolating discretization itself as the source of this gain rather than model capacity or Flow Matching. These results suggest that discretizing a robot's intent is a promising paradigm for scaling world models to real-time physical deployment. As our limitations discuss, realizing this promise more broadly will require representations expressive enough to avoid the failure modes a fixed-size codebook currently introduces, particularly for dynamic agents and finer-grained obstacle discrimination.

Reproducibility Statement

We have taken several steps to ensure the reproducibility of our results. Model weights and deployment code will be made publicly available upon publication (see footnote 1), including the code used for physical robot deployment on the Clearpath Jackal and Boston Dynamics Spot platforms. Complete architectural details, including the visual encoder, early-fusion strategy, discrete codebook configuration, and Flow Matching heads, are provided in Appendix D (Architecture and Implementation Details), with the full hyperparameter set reported in Table 9. The dataset composition is described alongside the training hyperparameters in the same section. Our Discrete Latent Planning procedure is fully specified in Algorithm 1 (Appendix F), including both sources of stochasticity (Site 1 and Site 2 noise) and their annealing schedules. Architectural design choices are empirically validated via the ablation studies in Appendix E (projection capacity, deterministic decoder heads) and Appendix B.4 (KPC cost term ablations, discrete manifold stability). Physical experiment protocols, including trial counts, starting orientation ranges, and success criteria, are described in Sec. 4.2.

AI USE STATEMENT

We used generative AI tools for several tasks: drafting initial versions of sections of this paper, which were subsequently edited and revised by the authors; checking grammar and readability; debugging code that the authors wrote. The authors have reviewed all AI-assisted work. Draft text was rewritten and verified by the authors against the paper's actual claims and results before inclusion; AI-suggested related work was independently verified by two authors as relevant and accurately characterized before citation; and AI-assisted code debugging was reviewed and tested by the three authors for correctness, with all code written by the authors themselves. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

References

Bo Ai, Stephen Tian, Haochen Shi, Yixuan Wang, Tobias Pfaff, Cheston Tan, Henrik I Christensen, Hao Su, Jiajun Wu, and Yunzhu Li. A review of learning-based dynamics models for robotic manipulation. *Science Robotics*, 10(106):eadt1497, 2025.

Arslan Ali, Junjie Bai, Maciej Bala, Yogesh Balaji, Aaron Blakeman, Tiffany Cai, Jiaxin Cao, Tianshi Cao, Elizabeth Cha, Yu-Wei Chao, et al. World simulation with video foundation models for physical ai. *arXiv preprint arXiv:2511.00062*, 2025.

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-Supervised Learning from Images with a Joint-Embedding Predictive Architecture, April 2023.

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.

Randall Balestriero and Yann LeCun. LeJEPA: Provable and Scalable Self-Supervised Learning Without the Heuristics, November 2025. URL http://arxiv.org/abs/2511.08544.

Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 15791–15801, 2025.

Adrien Bardes, Jean Ponce, and Yann LeCun. MC-JEPA: A Joint-Embedding Predictive Architecture for Self-Supervised Learning of Motion and Content Features, July 2023.

Daniel M Bear, Elias Wang, Damian Mrowca, Felix J Binder, Hsiao-Yu Fish Tung, RT Pramod, Cameron Holdaway, Sirui Tao, Kevin Smith, Fan-Yun Sun, et al. Physion: Evaluating physical prediction from vision in humans and machines. *arXiv preprint arXiv:2106.08261*, 2021.

Homanga Bharadhwaj, Debidatta Dwibedi, Abhinav Gupta, Shubham Tulsiani, Carl Doersch, Ted Xiao, Dhruv Shah, Fei Xia, Dorsa Sadigh, and Sean Kirmani. Gen2act: Human video generation in novel scenarios enables generalizable robot manipulation. In *Conference on Robot Learning*, pp. 3936–3951. PMLR, 2025.

Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023.

Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. 2024. URL https://openai.com/research/video-generation-models-as-world-simulators.

Sam Buchanan, Druv Pai, Peng Wang, and Yi Ma. *Learning deep representations of data distributions*. Online, August 2025.

Faris Chaudhry. The geometry of projection heads: Conditioning, invariance, and collapse. *arXiv preprint arXiv:2605.17180*, 2026.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*, pp. 1597–1607. PMLR, 2020.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. *Advances in neural information processing systems*, 36:9156–9172, 2023.

Kuan Fang, Patrick Yin, Ashvin Nair, Homer Rich Walke, Gengchen Yan, and Sergey Levine. Generalization with lossy affordances: Leveraging broad offline data for learning visuomotor tasks. In *Conference on Robot Learning*, pp. 106–117. PMLR, 2023.

Kartik Gupta, Thalaiyasingam Ajanthan, Anton van den Hengel, and Stephen Gould. Understanding and improving the role of projection head in self-supervised learning. *arXiv preprint arXiv:2212.11491*, 2022.

David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In *Advances in neural information processing systems*, volume 31. Curran Associates, Inc., 2018.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering Diverse Domains through World Models, January 2023.

Noriaki Hirose, Dhruv Shah, Ajay Sridhar, and Sergey Levine. SACSoN: Scalable Autonomous Data Collection for Social Navigation, June 2023.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Emiel Hoogeboom, Alexey A. Gritsenko, Jasmijn Bastings, Ben Poole, Rianne van den Berg, and Tim Salimans. Autoregressive diffusion models. In *International Conference on Learning Representations*, 2022.

Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. GAIA-1: A Generative World Model for Autonomous Driving, September 2023.

Hsiang-Wei Huang, Jianxu Shangguan, Junbin Lu, and Jenq-Neng Hwang. Leflow: Generative latent flow planning for world models. *arXiv preprint arXiv:2608.24855*, 2026.

Michael Janner, Yilun Du, Joshua B. Tenenbaum, and Sergey Levine. Planning with Diffusion for Flexible Behavior Synthesis, May 2022.

Li Jing, Pascal Vincent, Yann LeCun, and Yuandong Tian. Understanding dimensional collapse in contrastive self-supervised learning. *arXiv preprint arXiv:2110.09348*, 2021.

Haresh Karnan, Anirudh Nair, Xuesu Xiao, Garrett Warnell, Soeren Pirk, Alexander Toshev, Justin Hart, Joydeep Biswas, and Peter Stone. Socially Compliant Navigation Dataset (SCAND): A Large-Scale Dataset of Demonstrations for Social Navigation. *arXiv:2203.15041 [cs, eess]*, March 2022.

Rooholla Khorrambakht, Joaquim Ortiz-Haro, Joseph Amigo, Omar Mostafa, Daniel Dugas, Franziska Meier, and Ludovic Righetti. Worldplanner: Monte carlo tree search and mpc with action-conditioned visual world models. *arXiv preprint arXiv:2511.03077*, 2025.

Moo Jin Kim, Yihuai Gao, Tsung-Yi Lin, Yen-Chen Lin, Yunhao Ge, Grace Lam, Percy Liang, Shuran Song, Ming-Yu Liu, Chelsea Finn, and Jinwei Gu. Cosmos Policy: Fine-Tuning Video Models for Visuomotor Control and Planning, January 2026. URL http://arxiv.org/abs/2601.16163.

Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11523–11532, 2022.

Shuang Li, Yihuai Gao, Dorsa Sadigh, and Shuran Song. Unified video action model. *arXiv preprint arXiv:2503.00200*, 2025.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

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.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

Rosanne Liu, Joel Lehman, Piero Molino, Felipe Petroski Such, Eric Frank, Alex Sergeev, and Jason Yosinski. An intriguing failing of convolutional neural networks and the coordconv solution. *Advances in neural information processing systems*, 31, 2018.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow, September 2022. URL http://arxiv.org/abs/2209.03003.

Xinhao Liu, Jintong Li, Yicheng Jiang, Niranjan Sujay, Zhicheng Yang, Juexiao Zhang, John Abanes, Jing Zhang, and Chen Feng. Citywalker: Learning embodied urban navigation from web-scale videos. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 6875–6885, 2025.

Zhiheng Liu, Weiming Ren, Xiaoke Huang, Shoufa Chen, Tianhong Li, Mengzhao Chen, Yatai Ji, Sen He, Jonas Schult, Belinda Zeng, Tao Xiang, Wenhu Chen, Ping Luo, Luke Zettlemoyer, and Yuren Cong. Tuna-2: Pixel Embeddings Beat Vision Encoders for Multimodal Understanding and Generation, April 2026. URL http://arxiv.org/abs/2604.24763.

Pauline Luc, Natalia Neverova, Camille Couprie, Jakob Verbeek, and Yann LeCun. Predicting deeper into the future of semantic segmentation. In *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017. doi: 10.1109/ICCV.2017.77.

Pauline Luc, Camille Couprie, Yann LeCun, and Jakob Verbeek. Predicting future instance segmentation by forecasting convolutional features. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss (eds.), *Computer Vision – ECCV 2018*. Springer International Publishing, 2018. ISBN 978-3-030-01240-3.

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. Leworld-model: Stable end-to-end joint-embedding predictive architecture from pixels. *arXiv preprint arXiv:2603.19312*, 2026.

Yanghong Mei, Longteng Guo, Ming-Ming Yu, Guiyu Zhao, Xingjian He, and Jing Liu. Navwm: A unified navigation world model for foresight-driven planning. *arXiv preprint arXiv:2606.24101*, 2026.

Riccardo Mereu, Aidan Scannell, Yuxin Hou, Yi Zhao, Aditya Jitta, Antonio Dominguez, Luigi Acerbi, Amos Storkey, and Paul Chang. Generative world modelling for humanoids: 1x world model challenge technical report. *arXiv preprint arXiv:2510.07092*, 2025.

Vincent Micheli, Eloi Alonso, and François Fleuret. Transformers are Sample-Efficient World Models, March 2023.

Mohammad Nazeri, Anuj Pokhrel, Alexandyr Card, Aniket Datar, Garrett Warnell, and Xuesu Xiao. Vertiformer: A data-efficient multi-task transformer for off-road robot mobility. *arXiv preprint arXiv:2502.00543*, 2025.

E-Ro Nguyen, Yichi Zhang, Kanchana Ranasinghe, Xiang Li, and Michael S Ryoo. Pixel motion diffusion is what we need for robot control. *arXiv preprint arXiv:2509.22652*, 2025.

NVIDIA. Cosmos 3: Omnimodal World Models for Physical AI, June 2026. URL http://arxiv.org/abs/2606.02800.

NVIDIA, Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, Daniel Dworakowski, Jiaojiao Fan, Michele Fenzi, Francesco Ferroni, Sanja Fidler, Dieter Fox, Songwei Ge, Yunhao Ge, Jinwei Gu, Siddharth Gururani, Ethan He, Jiahui Huang, Jacob Huffman, Pooya Jannaty, Jingyi Jin, Seung Wook Kim, Gergely Klár, Grace Lam, Shiyi Lan, Laura Leal-Taixe, Anqi Li, Zhaoshuo Li, Chen-Hsuan Lin, Tsung-Yi Lin, Huan Ling, Ming-Yu Liu, Xian Liu, Alice Luo, Qianli Ma, Hanzi Mao, Kaichun Mo, Arsalan Mousavian, Seungjun Nah, Sriharsha Niverty, David Page, Despoina Paschalidou, Zeeshan Patel, Lindsey Pavao, Morteza Ramezanali, Fitsum Reda, Xiaowei Ren, Vasanth Rao Naik Sabavat, Ed Schmerling, Stella Shi, Bartosz Stefaniak, Shitao Tang, Lyne Tchapmi, Przemek Tredak, Wei-Cheng Tseng, Jibin Varghese, Hao Wang, Haoxiang Wang, Heng Wang, Ting-Chun Wang, Fangyin Wei, Xinyue Wei, Jay Zhangjie Wu, Jiashu Xu, Wei Yang, Lin Yen-Chen, Xiaohui Zeng, Yu Zeng, Jing Zhang, Qinsheng Zhang, Yuxuan Zhang, Qingqing Zhao, and Artur Zolkowski. Cosmos World Foundation Model Platform for Physical AI, January 2025.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An Open-Source Generalist Robot Policy, May 2024.

Chaoyi Pan, Giridharan Anantharaman, Nai-Chieh Huang, Claire Jin, Daniel Pfrommer, Chenyang Yuan, Frank Permenter, Guannan Qu, Nicholas Boffi, Guanya Shi, et al. Much ado about noising: Dispelling the myths of generative robotic control. In *International Conference on Learning Representations*, volume 2026, pp. 90575–90614, 2026.

Jiwon Park, Dongkyu Lee, I. Made Aswin Nahrendra, Jaeyoung Lim, and Hyun Myung. DreamFlow: Local Navigation Beyond Observation via Conditional Flow Matching in the Latent Space, March 2026. URL http://arxiv.org/abs/2603.02976.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4195–4205, 2023.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models. *arXiv:2112.10752 [cs]*, April 2022.

Reuven Y Rubinstein. Optimization of computer simulation models with rare events. *European Journal of Operational Research*, 99(1):89–112, 1997.

Lloyd Russell, Anthony Hu, Lorenzo Bertoni, George Fedoseev, Jamie Shotton, Elahe Arani, and Gianluca Corrado. Gaia-2: A controllable multi-view generative world model for autonomous driving. *arXiv preprint arXiv:2503.20523*, 2025.

Dhruv Shah, Benjamin Eysenbach, Nicholas Rhinehart, and Sergey Levine. Rapid exploration for open-world navigation with latent goal models. In *5th Annual Conference on Robot Learning*, 2021.

Dhruv Shah, Ajay Sridhar, Arjun Bhorkar, Noriaki Hirose, and Sergey Levine. GNM: A general navigation model to drive any robot. In *International Conference on Robotics and Automation (ICRA)*, 2023a.

Dhruv Shah, Ajay Sridhar, Nitish Dashora, Kyle Stachowicz, Kevin Black, Noriaki Hirose, and Sergey Levine. ViNT: A Foundation Model for Visual Navigation. arXiv, June 2023b.

Vlad Sobal, Wancong Zhang, Kynghyun Cho, Randall Balestriero, Tim G. J. Rudner, and Yann LeCun. Learning from Reward-Free Offline Data: A Case for Planning with Latent Dynamics Models, February 2025. URL http://arxiv.org/abs/2502.14819.

Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, November 2015.

Ajay Sridhar, Dhruv Shah, Catherine Glossop, and Sergey Levine. NoMaD: Goal Masked Diffusion Policies for Navigation and Exploration, October 2023.

Yue Su, Sijin Chen, Haixin Shi, Mingyu Liu, Zhengshen Zhang, Ningyuan Huang, Weiheng Zhong, Zhengbang Zhu, Yuxiao Liu, and Xihui Liu. World guidance: World modeling in condition space for action generation. *arXiv preprint arXiv:2602.22010*, 2026.

Jingwen Sun, Wenyao Zhang, Zekun Qi, Shaojie Ren, Zezhi Liu, Hanxin Zhu, Guangzhong Sun, Xin Jin, and Zhibo Chen. Vla-jepa: Enhancing vision-language-action model with latent world model. *arXiv preprint arXiv:2602.10098*, 2026.

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in neural information processing systems*, 33:7537–7547, 2020.

GigaBrain Team, Boyuan Wang, Bohan Li, Chaojun Ni, Guan Huang, Guosheng Zhao, Hao Li, Jie Li, Jindi Lv, Jingyu Liu, et al. Gigabrain-0.5 m*: a vla that learns from world model-based reinforcement learning. *arXiv preprint arXiv:2602.12099*, 2026a.

Robbyant Team, Zelin Gao, Qiuyu Wang, Yanhong Zeng, Jiapeng Zhu, Ka Leong Cheng, Yixuan Li, Hanlin Wang, Yinghao Xu, Shuailei Ma, et al. Advancing open-source world models. *arXiv preprint arXiv:2601.20540*, 2026b.

Stephen Tian, Chelsea Finn, and Jiajun Wu. A control-centric benchmark for video prediction. *arXiv preprint arXiv:2304.13723*, 2023.

Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *2015 ieee information theory workshop (itw)*, pp. 1–5. Ieee, 2015.

Samuel Triest, Matthew Sivaprakasam, Sean J Wang, Wenshan Wang, Aaron M Johnson, and Sebastian Scherer. Tartandrive: A large-scale dataset for learning off-road dynamics models. In *2022 International Conference on Robotics and Automation (ICRA)*, pp. 2546–2552. IEEE, 2022.

Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pp. 1096–1103. Association for Computing Machinery, 2008. ISBN 9781605582054. doi: 10.1145/1390156.1390294.

Youpeng Wen, Junfan Lin, Yi Zhu, Jianhua Han, Hang Xu, Shen Zhao, and Xiaodan Liang. Vidman: Exploiting implicit dynamics from video diffusion model for effective robot manipulation. *Advances in Neural Information Processing Systems*, 37:41051–41075, 2024.

Grady Williams, Andrew Aldrich, and Evangelos Theodorou. Model predictive path integral control using covariance variable importance sampling. *arXiv preprint arXiv:1509.01149*, 2015.

World Labs. Marble: A multimodal world model. https://www.worldlabs.ai/blog/marble-world-model, 2025.

Philipp Wu, Alejandro Escontrela, Danijar Hafner, Pieter Abbeel, and Ken Goldberg. Daydreamer: World models for physical robot learning. In *Conference on robot learning*, pp. 2226–2240. PMLR, 2023.

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.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, et al. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026.

Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation Alignment for Generation: Training Diffusion Transformers Is Easier Than You Think, October 2024.

Tianhe Yu, Ted Xiao, Austin Stone, Jonathan Tompson, Anthony Brohan, Su Wang, Jaspiar Singh, Clayton Tan, Dee M, Jodilyn Peralta, Brian Ichter, Karol Hausman, and Fei Xia. Scaling Robot Learning with Semantically Imagined Experience, February 2023.

Fatemeh Zargarbashi, Dhruv Agrawal, Jakob Buhmann, Martin Guay, Stelian Coros, and Robert W Sumner. Vq-style: Disentangling style and content in motion with residual quantized representations. *arXiv preprint arXiv:2602.02334*, 2026.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018.

Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

Ruijie Zheng, Jing Wang, Scott Reed, Johan Bjorck, Yu Fang, Fengyuan Hu, Joel Jang, Kaushil Kundalia, Zongyu Lin, Loïc Magne, et al. Flare: Robot learning with implicit world modeling. In *Conference on Robot Learning*, pp. 3952–3971. PMLR, 2025a.

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, 2025b.

Allan Zhou, Moo Jin Kim, Lirui Wang, Pete Florence, and Chelsea Finn. NeRF in the Palm of Your Hand: Corrective Augmentation for Robotics via Novel-View Synthesis, January 2023.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World Models on Pre-trained Visual Features enable Zero-shot Planning, November 2024.

Chuning Zhu, Raymond Yu, Siyuan Feng, Benjamin Burchfiel, Paarth Shah, and Abhishek Gupta. Unified world models: Coupling video and action diffusion for pretraining on large robotic datasets. *arXiv preprint arXiv:2504.02792*, 2025a.

Haoran Zhu, Zhenyuan Dong, Kristi Topollai, and Anna Choromanska. AD-L-JEPA: Self-Supervised Spatial World Models with Joint Embedding Predictive Architecture for Autonomous Driving with LiDAR Data, January 2025b.

APPENDIX

Table 4 summarizes how Hydra relates to prior navigation baselines across five dimensions – observation input, goal input, conditioning input, output modalities, and navigation policy type – extending the architectural comparison of Fig. 2.

Table 4: Comparison with visually-grounded robot navigation baselines. **I**: Image, **P**: Pose, **A**: Action.

Method	Obs. Input	Goal Input	Cond. Input	Output	Nav. Policy
GNM (Shah et al., 2023a)	I	I	I	P	Reactive
VINT (Shah et al., 2023b)	I	I	I	P	Reactive
NOMAD (Sridhar et al., 2023)	I	I	I	P	Reactive
VertiFormer (Nazeri et al., 2025)	I, A, P	I, P	A or P	I, A, P	Planning
NWM (Bar et al., 2025)	I	I	P	I	Planning
Hydra	I, A, P	I, P	A or P	I, A, P	Planning

A PROBLEM FORMULATION

Reactive Policy Formulation: Reactive foundation models (*e.g.*, ViNT (Shah et al., 2023b), GNM (Shah et al., 2023a), and NoMaD (Sridhar et al., 2023)) frame navigation as a direct sequence-to-sequence mapping problem. Given the observation history $\mathcal{H}_t$ and a goal representation $\mathcal{G}$ (typically a target image), the model f_π directly outputs a sequence of immediate future actions or spatial waypoints of length H:

$$\hat{a}_{t:t+H-1} = f_\pi(\mathcal{H}_t, \mathcal{G}). \tag{5}$$

While computationally lightweight and effective for local trajectory following, reactive policies lack foresight. They cannot predict the future visual state or consequences of their actions, making them susceptible to local minima and rendering them incapable of planning.

Visual World Model Formulation (Decoupled Planning): To overcome the myopia of reactive policies, WMs (*e.g.*, NWM (Bar et al., 2025)) frame navigation as a forward conditional dynamics task. Given the history $\mathcal{H}_t$ and a stochastically sampled sequence of future actions $a_{t:t+H-1}$ from a sampling-based planner, the model P_W generates the expected future visual states, $\hat{I}$:

$$\hat{I}_{t+1:t+H} \sim P_W(\cdot \mid \mathcal{H}_t, a_{t:t+H-1}). \tag{6}$$

This formulation requires Model Predictive Control (MPC) where the planner and WM operate on two separate manifolds. Because it predicts visual states conditioned on continuous actions, it relies on perceptual metrics to evaluate goal reaching, which is brittle because it does not consider the distance (Ai et al., 2025; Tian et al., 2023; Bear et al., 2021).

B EXTENDED RESULTS

B.1 SINGLE-STEP IMAGE QUALITY

Following NWM, we evaluate single-step visual predictive fidelity on a held-out subset of the RECON dataset. We benchmark Hydra against NWM/B and VertiFormer. As shown in Table 5, Hydra achieves competitive performance (DreamSim, LPIPS) despite a smaller parameter footprint and encoding multiple modalities with only 50 NFEs compared to NWM's 250 NFEs. While VertiFormer achieves the highest PSNR, this metric is notoriously misaligned with dynamic perception: deterministic models minimize MSE by averaging all possible futures, resulting in severe "ghosting" artifacts and blurry textures (Luc et al., 2017; 2018). PSNR strictly penalizes pixel-level noise but is blind to structural semantics, causing a higher score for VertiFormer. By leveraging Flow Matching over a discrete bottleneck, Hydra avoids regression-to-the-mean, preserving sharp, semantically valid structures and yielding better perceptual scores (LPIPS, DreamSim).

B.2 DATASET EVALUATION

Table 6 reports offline trajectory forecasting accuracy (ADE/FDE) on a held-out subset across three navigation datasets (SCAND, TartanDrive, and SACSoN); VertiFormer, a deterministic baseline,

Table 5: Offline single-step image quality on a held-out subset of the RECON dataset. Hydra achieves competitive accuracy with fewer parameters and NFEs.

Model	**Params**	DreamSim ↓	LPIPS ↓	PSNR ↑
NWM/B	200M	0.2480 ± 0.1265	0.5168 ± 0.1147	10.69
VertiFormer	27.11M	0.3738 ± 0.0560	0.5417 ± 0.0757	**18.29**
Hydra	143.29M	**0.1103** ± 0.0345	**0.2809** ± 0.0679	16.20

Table 6: Offline trajectory forecasting evaluation on a held-out subset across three navigation datasets (SCAND, TartanDrive, and SACSoN). Hydra achieves competitive policy accuracy while retaining the planning capability that reactive baselines lack.

Model	**Params**	SCAND		TartanDrive		SACSoN		Mean	
		ADE ↓	FDE ↓	ADE ↓	FDE ↓	ADE ↓	FDE ↓	ADE ↓	FDE ↓
GNM	8.6M	0.16	0.27	0.67	1.12	0.13	0.21	0.32	0.53
ViNT	29.6M	0.15	0.25	0.80	1.33	0.16	0.26	0.37	0.61
NoMaD	19.0M	0.24	0.43	1.311	2.35	0.22	0.39	0.59	1.06
VertiFormer	27.11M	**0.10**	**0.18**	**0.43**	**0.80**	**0.10**	**0.19**	**0.21**	**0.39**
Hydra	143.29M	0.12	0.20	0.48	0.85	0.12	0.21	0.24	0.42

edges out generative methods on these metrics. However, as reported in Table 3, offline performance does not reliably indicate real-world success. Offline evaluations inherently assume fixed trajectory lengths and guaranteed short-term goal reachability. Conversely, physical execution suffers from compounding centimeter-scale errors that progressively deviate the robot from its intended trajectory.

B.3 Long-Horizon Qualitative Comparison

Figure 4 shows the qualitative counterpart to the FVD comparison in Table 1 (Sec. 4.1). NWM initially renders high-fidelity frames, but its continuous latent space suffers from compounding errors; after approximately 8 seconds, the visual context rapidly degrades and decouples from the commanded actions (hallucination). Hydra's discrete intent manifold adheres accurately to the physical commands over the full 16-second horizon.

B.4 KPC Ablations

We ablate the individual terms of $\mathcal{J}_{\text{KPC}}$ (equation 2) to isolate which components drive Hydra's obstacle-avoidance behavior. All results are averaged over 10 physical trials per configuration.

Image-goal term. We first ask whether $\mathcal{C}_{\text{img}}$ should be active unconditionally. Table 7 compares two scenarios that differ only in whether the goal is visible from the ego camera: frontal obstacle avoidance, where the goal remains at least partially in view, and a blind corner turn, where the goal is entirely outside the camera view until after the turn. With $\mathcal{C}_{\text{img}}$ active in both cases, success rate drops from 80% when the goal is visible to 20% when it is not. Disabling the goal term when the goal is not visible ($\lambda_{\text{img}} = 0$) improves the success rate. Without a true visual anchor for the goal, the term can be minimized by any visually plausible candidate rather than the geometrically correct one, actively misleading the planner rather than aiding it. Based on this result, in the subsequent experiment (Table 8), $\mathcal{C}_{\text{img}}$ is disabled.

Table 7: Effect of the image-goal cost term (λ_{img}) conditioned on goal visibility. The same term that improves performance when the goal is visible degrades it when the goal is not, motivating conditioning $\mathcal{C}_{\text{img}}$ on goal visibility rather than applying it unconditionally.

Scenario	**SR (%)** ↑	**Time (s)** ↓
Frontal, $\lambda_{\text{img}} = 1$	80	24.7 ± 5.8
Blind corner, $\lambda_{\text{img}} = 1$	20	30.6 ± 12.1
Blind corner, $\lambda_{\text{img}} = 0$	70	34.5 ± 8.9

The remaining avoidance terms are individually necessary and embodiment-agnostic. With $\mathcal{C}_{\text{img}}$ handled as above, Table 8 ablates the three terms, $\mathcal{C}_{\text{prior}}$, $\mathcal{C}_{\text{vq}}$, and $\mathcal{C}_{\text{ent}}$, individually, on both Spot and



Figure 4: Comparison of 16-second video generation on a held-out subset of RECON. While NWM's continuous latents degrade and drift from the commanded actions over time, Hydra's discrete manifold maintains strict temporal grounding.

Table 8: Ablation of obstacle-avoidance cost terms (λ_{prior}, λ_{vq}, λ_{entropy}), across two robot embodiments: a blind corner turn (Spot) and frontal obstacle avoidance (Jackal). Each avoidance term degrades performance when removed on *both* platforms, indicating the mechanism is embodiment-agnostic.

	Spot		Jackal	
Configuration	**Time (s)** $\downarrow$	**Collisions** $\downarrow$	**Time (s)** $\downarrow$	**Collisions** $\downarrow$
Original (avoidance costs + $\mathcal{C}_{\text{geo}}$)	30.6 ± 12.1	$\mathbf{0.20 \pm 0.42}$	23.8 ± 0.39	$\mathbf{0.20 \pm 0.13}$
$\lambda_{\text{prior}} = 0$	33.9 ± 13.0	0.70 ± 0.48	21.2 ± 0.26	0.70 ± 0.59
$\lambda_{\text{vq}} = 0$	30.3 ± 10.0	0.50 ± 0.53	20.9 ± 0.34	0.60 ± 0.28
$\lambda_{\text{entropy}} = 0$	36.2 ± 18.9	0.40 ± 0.52	26.6 ± 0.45	0.80 ± 0.61

Jackal. Removing any single term increases collisions relative to the full cost on both platforms, with the effect even more pronounced on Jackal (0.20 rising as high as 0.80) than on Spot (0.20 rising to 0.40–0.70). That the same qualitative degradation appears on two robots with different kinodynamics indicates each term contributes a distinct, non-redundant safety signal.

Notably, $\mathcal{C}_{\text{prior}}$'s contribution follows from the unconditional expert prior being fit only to collision-free human demonstrations: a trajectory that intersects an obstacle is itself far outside this distribution. $\mathcal{C}_{\text{prior}}$ therefore penalizes collisions as a consequence of penalizing implausibility under the expert distribution, complementing $\mathcal{C}_{\text{vq}}$'s explicit OOD rejection in the visual codebook rather than duplicating it.

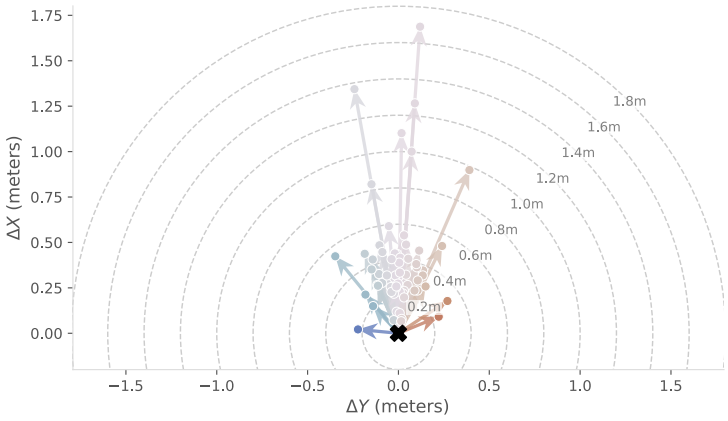


Figure 5: 1-step spatial coverage of the learned pose codebook $\mathcal{V}_p$. Unlike normalized baselines, the discrete manifold explicitly captures the absolute physical scaling of different robotic platforms, ranging from low-speed indoor maneuvers (dense central clusters) to high-speed off-road dynamics (sparse outer bounds).

B.5 Analysis of the Kinodynamic Codebook

While a continuous action space preserves infinite-resolution control, which is highly beneficial for fine-grained maneuvers if the transition model is perfect, it is sample-inefficient. Hydra mitigates the loss of fine-grained continuous resolution by using continuous Flow-Matching during final execution. These primitive discretized actions can be considered the mean of the trajectory decoded by the FM head. We analyze the 1-step spatial geometry of the learned discrete pose codebook, $\mathcal{V}_p$. As shown in Figure 5, the codebook captures a diverse, unnormalized distribution of spatial displacements $(\Delta X, \Delta Y)$.

On the other hand, a predefined or uniformly spaced grid of actions is fixed independently of the world model. Because such a grid is never trained jointly with the model's own representation, its entries have no defined relationship to the model's embedding space at all, a hand-specified "turn left" primitive is not a point the encoder ever produced, so there is no meaningful notion of its distance to anything the model has learned. By contrast, because Hydra's codebook is learned end-to-end alongside perception and dynamics, every entry is, by construction, a point the model itself produced and can situate relative to its own manifold. The distance metric between codebook entries is therefore meaningful only because the codebook and the world model share one coupled representation. This is the same measure of safety that the KPC cost uses to evaluate candidate trajectories in the latent space without decoding during the planning phase.

The Flaw of Action Normalization. Generalist navigation models (*e.g.*, GNM, ViNT, NoMaD, and NWM) universally normalize actions by the maximum speed of the specific robot platform to achieve dataset-agnostic training. However, this normalization destroys the absolute scale of the physical world; a maximum-throttle command for a slow indoor crawler becomes mathematically indistinguishable from a maximum-throttle command for a high-speed off-road ATV. Consequently, the generative model becomes blind to the physical constraints of the environment.

Emergent Physical Grounding. By using unnormalized actions, Hydra's codebook naturally segregates physical domains. Low-speed datasets populate the dense inner clusters of $\mathcal{V}_p$ ($< 0.5m$), while high-speed datasets form the sparse outer bounds ($> 1.0m$). Because the observation history $\mathcal{H}_t$ is contextually fused with these absolute velocities, Hydra learns a strict correlation between visual textures and physical speed limits.

This unnormalized grounding explains an emergent behavior in Hydra: *Contextual Manifold Shift*. If the model is conditioned on an indoor visual history (implying a slow-moving platform like a Clearpath Jackal) but is suddenly forced to execute a high-speed action intent (e.g., 1.5 m/step), the model will abruptly "hallucinate" a transition to an off-road TartanDrive environment. Rather than a standard generative failure, this is a resolution of conflicting modalities: the generative solver



Figure 6: Live RViz visualization of Hydra's Discrete Latent Planning search on the physical robot in a never-before-seen environment. **Left:** the robot's current ego-centric observation. **Middle:** a subset of Hydra's predicted future frames. **Right:** the corresponding candidate trajectories (green) rendered as a branching tree in the robot's local frame; each branch is one topological maneuver considered before $\mathcal{J}_{\text{KPC}}$ selects the lowest-cost (purple) candidate for execution.

recognizes that the commanded velocity has zero probability density within the indoor contextual manifold, and shifts the predicted visual flow to the only environmental sub-manifold where that velocity is physically valid.

By bounding trajectory searches to this physically grounded codebook (equation 2), Hydra prevents the execution of these OOD speed-context mismatches, ensuring safety in a way normalized models cannot. Furthermore, knowing which codes correspond to high-speed or turning, we can effectively change the robot's behavior during deployment by masking specific codes. For example, masking high-speed codes prevents the robot from driving fast, or masking right turns prevents the robot from turning right.

B.6 Visualizing Latent Search: RViz Branch Trees

To make Discrete Latent Planning's search process (Algorithm 1, Appendix F) inspectable, we visualize it in RViz during physical deployment.

Figure 6 shows a representative search: the search tree fans out from the robot's current pose into a small number of distinct trajectories reflecting the discrete codebook's role in constraining candidates to a small vocabulary of plausible maneuvers (Sec. B.5) rather than an unconstrained continuous spread.

B.7 Latent Collision Detection

To empirically examine Manifold Rejection ($\mathcal{C}_{vq}$) and Predictive Entropy ($\mathcal{C}_{ent}$) as complementary latent signals correlated with collision risk, we visualize both signals jointly for Hydra when forced into a geometric contradiction: a trajectory commanded to drive straight into an obstacle (Fig. 7).

We compute the VQ quantization error of the visual tokens and the Shannon entropy of their distribution at each rollout step. As the commanded trajectory approaches the obstacle, the continuous intent vector drifts toward the undefined space between valid discrete codebook anchors, and both signals rise in tandem: quantization error rises because the predicted embedding has no close codebook match, and entropy rises because the model becomes less confident about which future is plausible. When the generative head is ultimately forced to resolve this OOD state, a *Contextual Manifold Shift*, by hallucinating a navigable pavement in place of the occlusion, both signals drop together immediately afterward, because the fabricated resolution is once again close to something the model has seen and is confident about.

The signal is not perfectly discriminating, however. The smaller fluctuation in $\mathcal{C}_{vq}$ visible earlier in the rollout is a false positive: the model registers elevated quantization error while approaching visually complex but entirely traversable foliage, rather than a genuine obstacle. This introduces a limitation: dense, textured content such as leaves and grass can locally resemble the reconstruction difficulty of a true occlusion, even though no collision risk is present. Hydra's OOD signal is therefore a strong, but

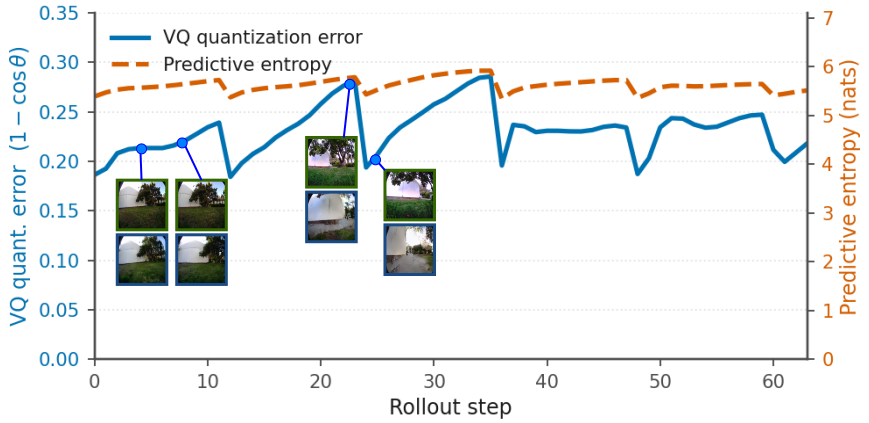


Figure 7: **Manifold Rejection and Predictive Entropy During Impending Collision.** Top/Bottom insets: Comparison of Ground Truth vs. Hydra's imagination at points along a rollout commanded to drive straight toward a physical occlusion. Both the spatially-averaged VQ quantization error ($\mathcal{C}_{vq}$, solid blue, left axis) and the predictive entropy of the visual token distribution ($\mathcal{C}_{ent}$, dashed orange, right axis) rise together as the commanded trajectory approaches the occlusion, and drop together immediately after the model hallucinates a navigable resolution, two independently computed signals converging on the same collision-relevant moments. The smaller early fluctuation (first two insets) is a false positive triggered by visually complex but traversable foliage, not a genuine obstacle.

not infallible, indicator of collision risk. The reverse failure is also possible: homogeneous obstacles such as flat walls are comparatively easy to reconstruct and may therefore be under-flagged despite being genuine hazards. Because the present signal cannot separate *why* a state is hard to reconstruct, occasional false positives and potentially false negatives are unavoidable with the current architecture. Addressing this will likely require a representation that distinguishes structural novelty that should be avoided from complex visual texture that is safe to traverse, rather than collapsing both into a single quantization-error signal.

This joint behavior is nonetheless the crux of Hydra's DLP. Because $\mathcal{C}_{vq}$ and $\mathcal{C}_{ent}$ are computed independently, one from quantization geometry, one from predictive uncertainty, but rise and fall together at exactly the moments that matter, heavily weighting both in the KPC (equation 2) lets the optimizer reject a candidate trajectory *before* the generative head is forced to hallucinate, using two agreeing, pixel-free signals rather than one. Occasional false positives aside, this remains substantially more reliable than the alternative of not evaluating safety at all until pixels are rendered: the entire mechanism, ranking, and rejection, happens in latent space, extracting a safety constraint directly from the quantization and uncertainty of the visual world model without ever decoding to pixels.

B.8 Failure Analysis

Failure mode: missed pedestrians. Figure 8 shows a representative failure mode on SCAND's social navigation scenarios: pedestrians present in the ground-truth future are absent from Hydra's prediction. We attribute this to the limited cardinality of the visual codebook ($|\mathcal{V}_i| = 2048$), which acts as a stochastic low-pass filter: humans occupy a small subset of pixels relative to the static scene, and are discarded as high-frequency detail during quantization, the same limitation noted in our main-text discussion of limitations. This issue exists in NWM as well, but at a lower frequency. Incorporating human-centric object detection or increasing visual codebook capacity are natural directions to address this in future work.

Failure mode: goal-blind seed initialization. In Fig. 9, the green trajectory shows the path selected by the DLP cost among the sampled candidates. This failure arises from seed conditioning: the seed distribution is generated without knowledge of the goal, so goal information only enters the



Figure 8: Vanishing humans in future predictions on SCAND. Top: ground-truth future frames, in which a pedestrian remains visible throughout. Bottom: Hydra's predicted frames for the same sequence, in which the pedestrian is smoothed into the background.

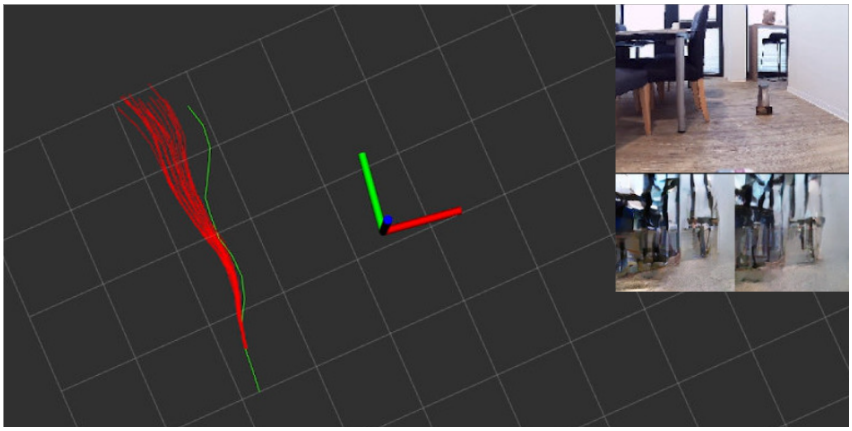


Figure 9: Goal-blind seed initialization causes a missed turn. The green trajectory is the path selected by the DLP cost among the sampled candidates at convergence; the seed distribution that generated the candidates was conditioned only on local scene context, not on the goal, delaying goal-aware correction until it is too late to complete the turn.

optimization indirectly, through the Euclidean-distance term of the KPC cost, which is evaluated *after* each iteration's candidates are scored and only biases the seed population for the *next* iteration. With $N = 3$ iterations, set by the real-time planning budget, this leaves little runway for goal information to propagate before the robot must act. When a required turn falls close to the start of the planning horizon, the robot can physically pass the turning point before the distance signal has had enough iterations to steer the search toward it.

Design tradeoff: goal-conditioned vs. unbiased seeding. This failure mode reflects a more general tension between convergence speed and search diversity. We conditioned the seed on the goal direction. This resolved the near-turn failure directly, since the search no longer needed multiple iterations to discover the goal direction. However, it introduced a different problem: with all seeds already pointed at the goal, the sampled candidates clustered close together in trajectory space, reducing the diversity of the search and limiting exploration of the KPC cost landscape. We chose unbiased seed conditioning as the more conservative choice, accepting the occasional near-turn failure mode described above in exchange for preserving exploration diversity across the sample population. We leave resolving this tradeoff, for instance via coarser goal-direction (rather than full goal-pose) conditioning, or an explicit diversity term applied alongside goal-conditioning, to future work.

C Learning Perspective on Hydra

As shown in Figure 1, Hydra moves planning from an external module that queries a separately trained world model to a planner that lives inside the world model. We discuss the different parts of Hydra from a learning perspective that aided this transition.

Joint Distribution vs. Marginal Alignment. From a representation learning perspective (Buchanan et al., 2025), traditional late-fusion architectures optimize isolated marginal distributions and attempt to align them post-hoc (Liu et al., 2023) to prevent the dominant modality's gradients from overcoming the other (Liu et al., 2026), often requiring complex contrastive losses. TUNA-2 (Liu et al., 2026) showed that this causes representation misalignment, however, given a large amount of data, the model can learn to align different modalities end-to-end. VertiFormer (Nazeri et al., 2025) showed that, by employing early fusion, a model is able to approximate the joint multimodal distribution $p(I, p, a)$ on limited data. This imposes an inductive bias on the Transformer to learn the underlying physical correlations (*e.g.*, forward kinematics and inverse dynamics) from the unified token sequence. Furthermore, the curriculum-based random infilling acts as a generalized Denoising Autoencoder objective (Vincent et al., 2008), aiding the network in learning representations that smoothly interpolate across missing sensor modalities.

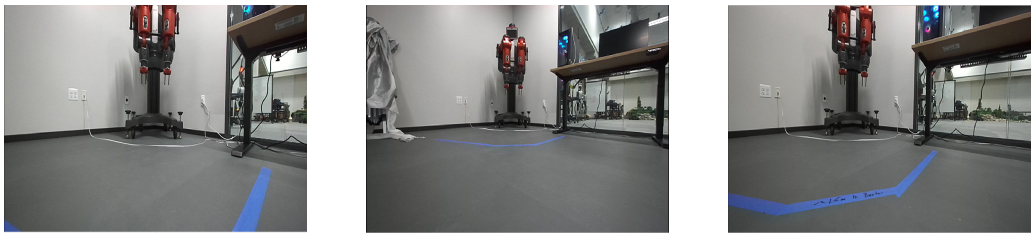
Information Bottleneck and Posterior Collapse. The necessity of the discrete VQ bottleneck can be analyzed through the lens of the Information Bottleneck principle (Tishby & Zaslavsky, 2015). Unbounded continuous latent spaces in autoregressive or generative sequence models are notoriously prone to posterior collapse or high-frequency hallucination, as the model can perfectly memorize noisy inputs without extracting underlying causal structure. By forcing the continuous intent through a fixed-size VQ codebook, we bound the mutual information between the observation history and the generated future. This information constraint forces the network to discard accumulated low-level motor noise between prediction steps and compress interactive dynamics into a compact manifold of semantically meaningful kinodynamic primitives.

With this objective in mind, because these codebook embeddings are initialized randomly, passing them directly into an uncalibrated continuous differential equation solver causes rapid divergence. Therefore, we introduce a deterministic linear anchoring that ensures the codebook manifold remains stable (Jing et al., 2021) while the downstream generative ODE learns to map this intent to continuous trajectories. Before the flow-matching stage, we apply an MSE loss to align the transformer output and the embedding of the ground truth modality. This is similar to pretraining the model on next-frame prediction, then fine-tuning it to predict high-quality images using flow matching; however, we do both stages at the same time, mimicking the REPA (Yu et al., 2024) regularization term.

A deep, non-linear projection would possess sufficient capacity to untangle a disorganized latent space (Chen et al., 2020), inadvertently permitting the VQ codebook to remain uninformative and unstructured, whereas a linear projection forces implicit subspace whitening on the discrete codebook (Chaudhry, 2026; Gupta et al., 2022). Furthermore, a linear projection mathematically compels the discrete embeddings to construct a highly structured, linearly separable manifold early in the training process. This constrained 'tight grip' ensures that the conditioning signal provided to the subsequent Flow Matching module acts as a robust, geometrically meaningful prior. We empirically validate the necessity of this strict linear information bottleneck in Appendix E.1, demonstrating that it significantly improves codebook utilization and training convergence.

Optimal Transport and ODE Stability. While recent generative policies heavily rely on score-based Diffusion Models (Chi et al., 2025), diffusion is formulated as a Stochastic Differential Equation (SDE) that suffers from highly curved sampling trajectories. Hydra utilizes Rectified Flow Matching (Liu et al., 2022), which frames the generative process as an Ordinary Differential Equation (ODE). From an optimization standpoint, mapping a Gaussian prior directly to the target distribution via a linear velocity field $u(\psi_s) = \psi_1 - \psi_0$ implicitly encourages Optimal Transport (OT) (Lipman et al., 2022). This OT property results in straighter, deterministic probability paths during inference. For robotics, this is a critical theoretical advantage: straighter ODE paths allow the model to be integrated with significantly fewer Number of Function Evaluations (NFEs) without discretization error, enabling the real-time inference speeds required for physical deployment. This is in contrast with NWM, which formulates the generations as an SDE.

The Perceptual Fallacy: Limitations of LPIPS. Prior image-goal navigation frameworks utilizing world models often rely on visual similarity metrics to evaluate predicted trajectories. NWM, for instance, utilizes Learned Perceptual Image Patch Similarity (LPIPS) to calculate the distance between the predicted visual future and the goal image.



(a) Goal Image (b) LPIPS Loss: 0.50 (c) LPIPS Loss: 0.52

Figure 10: LPIPS deviation from geometric grounding. Note that these are not generated images.

D Architecture and Implementation Details

The architecture is modularly separated into the visual-proprioceptive encoders, the unified TransformerEncoder, the discrete codebooks, and the multimodal latent flow decoders.

Visual Encoder. To compress high-resolution visual observations $I_t \in \mathbb{R}^{3\times224\times224}$ into spatial latents, we used the pretrained Stable Diffusion AutoencoderKL (Blattmann et al., 2023), which produces latents $z_i \in \mathbb{R}^{4\times28\times28}$, and kept it frozen during training.

Infilling Curriculum Strategy. To ensure robustness against partial observability during deployment (*e.g.*, missing telemetry) and to induce unconditioned exploration, we apply a learned modality infilling strategy introduced by Nazeri et al. (2025) during training. Instead of employing a naive hard-switching dropout strategy (Nazeri et al., 2025), we implement a cosine-annealed modality infilling scheduler. During the initial epochs, the network is trained on the fully observable joint distribution. As training progresses, the masking probability for pose and action tokens is smoothly increased following a cosine curve, preventing gradient shock and teaching the early-fusion Transformer to infer missing modalities without destabilizing the latent flow.

Over the first $E_{warmup} = 10$ epochs, we gradually decay the probability of the network receiving all modalities (the fully observable joint distribution, p_{joint}) from 0.5 to 0.1. For a given epoch e, the probabilities are defined as:

$$p_{joint} = 0.1 + 0.4\left[\frac{1}{2}\left(1+\cos\left(\pi\min\left(1,\frac{e-1}{E_{warmup}}\right)\right)\right)\right] \tag{7}$$

The remaining probability mass is split evenly between the FKD mask state and the IKD mask state:

$$p_{fkd} = p_{ikd} = \frac{1-p_{joint}}{2} \tag{8}$$

This soft-switching curriculum helps the early-fusion Transformer to learn the fully observable transition dynamics before being forced to infer missing marginals.

Early Fusion. Encoded actions and poses are broadcasted to a uniform length and summed to create a single, unified token stream (Li et al., 2025). To prevent receiving duplicated gradients for actions and poses due to broadcasting, which causes the model's over-sensitivity to actions and poses, we scaled down the actions and poses by the square root of their length, $\sqrt{k}$, where k is the length of the broadcasted sequence. To early fuse modalities before passing them to the Transformer, Hydra simply sums all the modalities to create a unified representation without introducing extra parameters, and then adds sinusoidal positional encoding (Vaswani et al., 2017). This simple modulation works and cuts unnecessary parameters.

Flow Matching Head. To generate continuous trajectories and high-fidelity visual latents from the discrete intent vectors, Hydra employs a conditional Rectified Flow Matching (Liu et al., 2022)

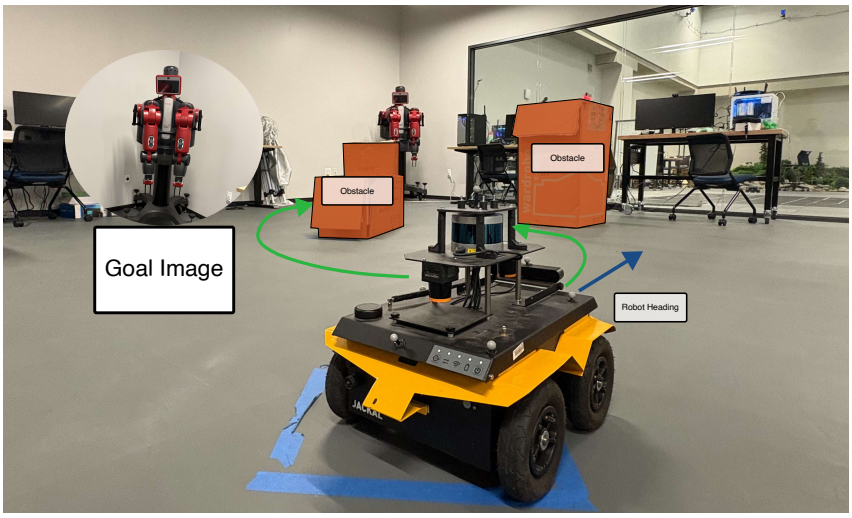


Figure 11: Physical experiment setup with obstacles.

framework modulated via AdaLN-Zero. We deliberately tailor the representational capacity of the Flow Matching network to the specific spatial-temporal and latency constraints of each modality.

For the high-dimensional visual world model, we utilize a full Diffusion Transformer (DiT) block (Peebles & Xie, 2023) with coordinate channels (Liu et al., 2018) and a hidden dimension of 768, allowing spatial patches to continuously query each other via Multi-Head Self-Attention (MHSA) to maintain structural coherence across time. On the other hand, the DiT block for action and pose Flow Matching heads reduces the hidden dimension to 256 to achieve faster inference.

Datasets. To train Hydra, we utilize the same heterogeneous robotic dataset corpus (Shah et al., 2023a) as NWM. It contains 377,580 samples, broken down into 592,045,440 tokens seen during training. This corpus encompasses a diverse set of environments, including off-road (Triest et al., 2022), indoor (Hirose et al., 2023), outdoor (Shah et al., 2021), and social (Karnan et al., 2022) navigation scenarios. The dataset is structured as synchronized temporal tuples of $\{I_t, p_t, a_t\}$, providing the structural alignment required to jointly optimize Hydra's early-fusion latent manifold across visual, spatial, and kinematic domains. Unlike NWM and GNM, we *do not* standardize the step size across agents by dividing the distance agents travel between frames by their maximum step size in meters, so that the model can discriminate between lower-speed robots like Clearpath Jackal and high-speed platforms such as the ATV used in TartanDrive; instead, we rely on the codebooks to capture these differences, as shown in Sec. B.5. We filter out backward movements, following NWM and GNM.

Training Hyperparameters. The model is trained end-to-end (with the AutoencoderKL frozen) using the AdamW optimizer. Table 9 reports the complete hyperparameter set.

Variable Target Horizon. While the maximum prediction horizon is fixed to 3 seconds ($H = 12$ steps at 4Hz), we utilize a variable-length temporal regularization strategy during training. For each batch, the target rollout length is uniformly sampled between 1 and 3 seconds. This prevents the model from overfitting to a rigid temporal boundary, ensuring robust transition dynamics at arbitrary planning horizons during physical deployment.

E Architectural Ablations

To isolate the specific contributions of Hydra's architectural components, we perform a series of ablation studies focusing on the discrete intent bottleneck, context compression, and inference memory constraints.

Table 9: Hyperparameters and architectural details of Hydra. The model utilizes a Transformer backbone combined with dimensionally constrained, modality-specific VQ codebooks and Flow Matching decoders.

Parameter	Value
Training & Optimization	
Optimizer	AdamW
Learning Rate	2×10^{-4}
Weight Decay	0.08
Batch Size	8
Gradient Accumulation Steps	32
Total Effective Batch Size	256
Epochs	60
Gradient Clipping	1.0
Variable Prediction Horizon	$4 \ldots 12$ steps
Dataset Parameters	
Observation Length (H_{obs})	8 steps
Prediction Length (H_{pred})	12 steps
Image Size	224×224
Patch Size	2×2
Unified Transformer Architecture	
Visual Backbone	SD-VAE-FT-EMA
Positional Encoding	Sinusoidal 3D
Transformer Layers (Encoder / Decoder)	6 / 6
Transformer Dimension (d_{model})	768
Attention Heads	12
Feedforward Dimension	2048
Dropout	0.1
Vector Quantization (VQ) Bottleneck	
Image Codebook Size ($\mathcal{V}_i$)	2048
Action Codebook Size ($\mathcal{V}_a$)	64
Pose Codebook Size ($\mathcal{V}_p$)	64
Codebook Dimension	768
Commitment Cost (β)	0.25
EMA Decay (γ)	0.99
Flow Matching Heads	
ODE Block Architecture	DiT block
Hidden Dimension, Image Head	768
Hidden Dimension, Pose/Action Heads	256
Sampling Steps, Image Head (NFEs)	50
Sampling Steps, Pose/Action Heads	10
ODE Source Distribution	$\mathcal{N}(0, \mathbf{I})$
Deterministic Loss Weight	0.1

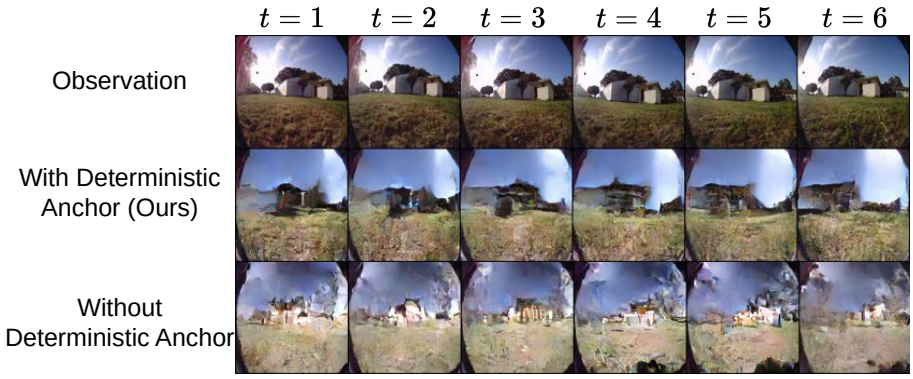


Figure 12: Early Training Dynamics (2.1K Steps). A comparison between the model's early-stage generation, both with (middle) and without (bottom) the deterministic anchor head. While the bottom row suffers from spatial smearing, the model utilizing the deterministic head shows a better ability to construct structural details, like the building's edges against the sky, even after very limited training.

E.1 Projection Capacity and Codebook Utilization

To empirically validate the necessity of the strict linear information bottleneck preceding the Flow Matching solver, we ablated the projection layer architecture. We compared our standard linear projection against a higher-capacity 2-layer MLP with GeLU activation. We evaluated the models at 237K training steps, analyzing both codebook perplexity (the entropy of discrete code utilization) and the cosine similarity of the multi-task gradients.

While both architectures achieved 100% code usage, the linear projection yielded a significantly higher image codebook perplexity (608.36) compared to the MLP (465.42). This indicates that the non-linear MLP possesses enough capacity to lazily route information, resulting in an unstructured, low-entropy utilization of the visual manifold. Conversely, the strict linear projection forces the network to uniformly distribute representations across the available codebook.

E.2 Deterministic Decoder Heads

Besides the projection capacity, to empirically validate the necessity of this deterministic projection preceding the VQ codebooks, we visualized the early training dynamics (at 2.1K steps) with and without this auxiliary head. As shown in Figure 12, passing randomly initialized codebooks directly into the continuous flow-matching solver causes severe representational instability. Conversely, incorporating the deterministic head provides an immediate stabilizing prior. By grounding the codebooks with a standard MSE loss, the network is forced to maintain spatial coherence early in the training process, preventing the downstream generative ODE from diverging and accelerating the formation of structural scene details.

F Latent Planning Algorithm Details

F.1 Two Sources of Stochasticity

Generating diverse candidate trajectories from a discrete VQ latent space requires randomizing two different things, at two different stages of the search.

Why codebook-level noise alone is not enough. A natural first attempt is to compute the Transformer's intent tokens once, then let each of the B candidates independently sample a codebook entry from the resulting logits. This produces candidates that differ only in *which nearby codebook vector each one snaps to*, and because the codebook is trained so that nearby vectors are semantically similar, swapping the 1st-nearest for the 3rd-nearest vector rarely changes the decoded trajectory's shape in any meaningful way. This gives quantization jitter around a single underlying idea (*e.g.*,

"swerve left" vs. "swerve left, very slightly differently"), never qualitatively different intents (*e.g.*, "swerve left" vs. "swerve right").

Site 1: Gaussian noise on the continuous intent query. To obtain genuinely different trajectory shapes, the input to the Transformer decoder itself must change. Since the Transformer, not the codebook, decides the high-level shape of the candidate we therefore perturb the continuous intent query independently for each candidate before it is passed to the decoder:

$$\tilde{\epsilon}_b[t] = \alpha_{sm} \cdot \epsilon_b[t] + (1 - \alpha_{sm}) \cdot \tilde{\epsilon}_b[t-1], \qquad \epsilon_b \sim \mathcal{N}(0, I), \tag{9}$$

smoothed across the horizon via an Exponential Moving Average (EMA) with coefficient α_{sm} so that a perturbation at one timestep biases neighboring timesteps in a similar direction, producing a smooth alternate path (*e.g.*, "curve further right the whole way") rather than step-to-step jitter that would be physically implausible for a robot to execute. Each candidate's noisy query, $q_b^{(i)} = P_{intent}^{(i-1)} + \sigma^{(i)}\tilde{\epsilon}_b$, is then passed through a fresh Transformer forward pass, yielding B genuinely different sets of intent tokens rather than B discretizations of one shared set. The noise scale $\sigma^{(i)}$ shrinks geometrically across iterations, giving the search a coarse-to-fine, CEM-like character: broad exploration early, local refinement late.

Site 2: Top-K Gumbel-max discretization. Each candidate's (now distinct) continuous intent tokens must still be snapped onto valid codebook vectors before decoding. We do this stochastically, restricted to the K nearest vectors by cosine similarity, rather than with a hard nearest-neighbor snap, adding a second, finer layer of exploration around each of Site 1's hypotheses while the top-K restriction keeps candidates from drifting to codebook entries that are cosine-far, *i.e.*, off-manifold and physically implausible (Sec. B.5). Like Site 1, the raw Gumbel noise driving this discretization is itself temporally smoothed via its own EMA, with its own coefficient α_{gum}, independent of Site 1's α_{sm}: without this, independent per-timestep Gumbel draws would let each candidate's discretization jitter step to step even when its underlying continuous trajectory (from Site 1) is smooth. This is detailed in the remainder of this section.

Both sites are necessary and serve different purposes: Site 1 diversifies *what the model considers*; Site 2 makes each consideration's discretization robust rather than collapsed onto a single nearest neighbor, while remaining temporally coherent in its own right. Algorithm 1 uses both simultaneously, each independently smoothed and annealed.

Algorithm 1 Discrete Latent Planning (Hydra)

Require: Observation history $\mathcal{H}_t$, goal pose P_{goal}, visual goal z_{goal}, horizon T, candidates B, iterations N

$h_{ctx} \leftarrow$ UnifiedEncoder($\mathcal{H}_t$) ▷ encode history once
$P^{(0)} \leftarrow$ unbiased expert prior pose sequence, broadcast to B candidates
$J_{best} \leftarrow \infty$, $c_{best} \leftarrow \emptyset$
for $i = 0$ to $N - 1$ **do**
  if $i > 0$ **then**
    $P^{(i)} \leftarrow P^{(i-1)}+$ decaying smoothed Gaussian noise, per candidate ▷ explore pose space
  end if
  $z_e \leftarrow$ TransformerDecoder($h_{ctx}, P^{(i)}$)
  $c \leftarrow$ discretize z_e via Top-K-restricted, annealed Gumbel-Softmax sampling
  $\hat{P}, \hat{z}_{img} \leftarrow$ decode c to pose and visual codes
  $J \leftarrow$ KPC($\hat{P}, \hat{z}_{img}, P_{goal}, z_{goal}$) ▷ Eq. equation 2, per candidate
  $b^* \leftarrow \arg\min_b J_b$
  if $J_{b^*} < J_{best}$ **then**
    $J_{best} \leftarrow J_{b^*}$; $c_{best} \leftarrow c_{b^*}$
  end if
  $P^{(i+1)} \leftarrow$ momentum update toward $\hat{P}_{b^*}$, broadcast to B candidates ▷ shift search region
end for
return FlowDecoder(Lookup($c_{best}, \mathcal{V}_p$)) ▷ high-fidelity continuous execution

Note that J_{best} and c_{best} track the globally best candidate across *all* N iterations.

Standard Gumbel-Max Trick. Let $s_{t,c}$ denote the unnormalized logit predicted by the Transformer decoder for timestep $t \in \{1, \ldots, T\}$ and codebook entry $c \in \mathcal{V}_p$. Sampling from the Categorical

distribution $p(c_t) \propto \exp(s_{t,c}/\rho)$ with temperature ρ is equivalent to the Gumbel-Max trick:

$$c_t = \arg\max_{c \in \mathcal{V}} \left(\frac{s_{t,c}}{\rho} + g_{t,c} \right)$$

where $g_{t,c} \sim \text{Gumbel}(0, 1)$ is i.i.d. noise sampled via $g_{t,c} = -\log(-\log(u_{t,c}))$ with $u_{t,c} \sim \mathcal{U}(0, 1)$.

Iteration-Dependent Annealing Schedule. Since the search is an iterative optimizer running for N depth iterations, we anneal the Site 2 temperature ρ and Site 2's own smoothing coefficient α_{gum} at each iteration i. Site 1's smoothing coefficient α_{sm} is, by contrast, a fixed constant and is not annealed; only its noise *scale* $\sigma^{(i)}$ shrinks across iterations.

Simulated Annealing (Coarse-to-Fine Search): We exponentially decay the Site 2 temperature to transition from broad topological exploration to greedy exploitation:

$$\rho^{(i)} = \rho_0 \cdot 0.1^{\,i}$$

where ρ_0 is the base temperature.

Annealing the Gumbel Smoothing Factor: We linearly increase α_{gum} as iterations progress:

$$\alpha_{gum}^{(i)} = \min\left(0.4 + 0.15 \cdot i,\ 1.0\right)$$

During early iterations, $\alpha_{gum}^{(i)}$ is small, generating highly correlated Gumbel noise for large-scale, smooth exploration; in late iterations, $\alpha_{gum}^{(i)} \to 1.0$, allowing fine-grained, near-independent micro-adjustments to individual timesteps along the elite trajectory.

Top-K Kinematic Masking. Even with temporally correlated exploration, the planner might occasionally sample outlier codes that violate kinematic constraints. We strictly bound the sampling space by projecting a hard constraint mask over the logits. Let $\mathcal{T}_t$ be the set containing the indices of the K largest logits at timestep t. We define the mask $\mathbf{M} \in \{0, -\infty\}$ as:

$$M_{t,c} = \begin{cases} 0 & \text{if } c \in \mathcal{T}_t \\ -\infty & \text{otherwise} \end{cases}$$

This formulation ensures that the sampled latent intent remains within the physically plausible manifold learned by Hydra (enforced by $\mathbf{M}$), while the Site 1 perturbation ensures that the sampled intents exhibit the temporal coherence necessary for stable robotic control. Table 10 shows the DLP hyperparameters used during experiments.

Table 10: DLP Hyperparameters.

λ_{img}	λ_{prior}	λ_{ent}	λ_{vq}	γ	**# Candidates**	**Iterations**	top_k **samples**
0.02	0.5	0.5	1.0	0.85	18	3	10

F.2 Classifier-Free Guidance for Navigation Trajectory Generation

Vocabulary-based Maneuver Candidate Generation. To circumvent the OOD hallucination risk of latent extrapolation, we restrict the search space to the valid pose codebook $\mathcal{V}_p$. We observe that the codebook itself encodes a diverse maneuver vocabulary (Fig. 5): its entries, decoded as intents, span endpoint headings covering both turn signs. Given the mean-pooled unconditional prior $\bar{z}_\emptyset = \frac{1}{K}\sum_{k=1}^{K} z_\emptyset^{(k)}$, where K is an arbitrary number, we query the model unconditionally to get a more stable intent because of the randomness in FM heads. Then, we construct a batch of N candidate intents in a single forward pass as:

$$\mathcal{C} = \underbrace{\{\bar{z}_\emptyset\}}_{\text{expert default}} \cup \underbrace{\{\mathbf{e}_c\}_{c \in \mathcal{V}_p}}_{\text{pure maneuvers}} \cup \underbrace{\{\bar{z}_\emptyset + \lambda_j(\mathbf{e}_{c_j} - \bar{z}_\emptyset)\}}_{\text{convex mixes}}, \quad \lambda_j \sim \mathcal{U}[0.25, 1], \tag{10}$$

where $\mathbf{e}_c$ denotes codebook embedding c, broadcast as a constant intent over the horizon, and the candidate budget is split evenly between pure codebook entries and convex mixes that interpolate

between the scene-conditioned prior and the vocabulary. Importantly, every candidate is re-quantized through the VQ bottleneck before decoding, so even the mixes are snapped back onto the discrete manifold; by construction only valid codes are ever decoded. Where stochastic diversity is desired, we add small Gaussian perturbations drawn *once per candidate and timestep*, shared across the H intent slots and smoothed over time by an exponential moving average. Sharing the noise across slots is essential.

Deterministic Intent Decoding. Candidates are decoded with the deterministic pose head D_{pose}, App. E.2), rather than the flow-matching head. The deterministic head is pointwise stable, which makes candidate scores comparable across the batch, and it is trained on the intents it receives here. The stochastic ODE decoder, in contrast, exhibits high per-seed variance for maneuver-code candidates that lie far from the prior mode, which would render candidate comparison difficult. The same decode path is then used to execute the winner, so selection and execution are identical by construction. Since the deterministic decode is a single matrix multiplication per candidate, scoring N hypotheses is cheaper than a single flow-matching integration, preserving the low latency required for real-time control.

F.3 Sampling: Candidate Path Guidance

When the global planner supplies a reference path $\mathcal{G} = \{g_1, \dots, g_s\}$, we therefore score each candidate with a truncated Chamfer distance to $\mathcal{G}$. Let $\ell_t = \sum_{i \le t} \|p_i - p_{i-1}\|_2$ be the candidate's arc length and $L_{\mathcal{G}}$ the total arc length of the reference. We retain only the prefix $\mathcal{K} = \{t : \ell_t \le L_{\mathcal{G}}\}$ and compute:

$$d_{\text{path}}(\tau) = \frac{1}{|\mathcal{K}|} \sum_{t \in \mathcal{K}} \min_s \|\tau_t - g_s\|_2 \;-\; w_{\text{prog}} \cdot \rho_{\text{reward}}(\tau), \qquad \rho_{\text{reward}}(\tau) = \frac{\arg\min_s \|\tau_T - g_s\|_2}{S-1}, \tag{11}$$

where each candidate point is matched to its *nearest* reference point, so candidates are not penalized for executing the path at a different speed than the reference, and $\rho_{\text{reward}}(\tau)$ is a progress reward measuring how far along $\mathcal{G}$ the candidate reaches. The path cost is blended with the candidate's deviation from the unconditional expert trajectory $p^{\text{prior}}_{1:T}$:

$$\mathcal{J}_{\text{track}}(\tau) = \omega \cdot d_{\text{path}}(\tau) \;+\; \max(1-\omega,\, 0) \cdot \underbrace{\frac{1}{T} \sum_t \|p_t - p_t^{\text{prior}}\|_2}_{\text{prior deviation}}, \tag{12}$$

where $\omega{=}1$ yields strict path following and $\omega{<}1$ increasingly trusts the obstacle-avoiding prior. While $\omega{>}1$ is permitted, it reduces the second term to zero, amplifying the path signal. The selected intent is decoded to continuous actions and executed directly.

Computational Trade-offs. By restricting candidate generation exclusively to the unbiased expert prior, every sampled trajectory lies on the learned manifold. While evaluating visual predictive entropy or manifold rejection over these candidates (as done in Sec. 3.3) would provide obstacle avoidance, we explicitly omit the image codebook and decoder from both local sampling approaches. This omission is a strategic pruning choice designed to maintain low inference latency, relying entirely on the inherent safety of the expert before handling local obstacle avoidance. The primary limitation of this method occurs in highly constrained environments (*e.g.*, sharp U-turns), where the unconditional prior may assign near-zero probability to the necessary maneuver, relying on the diversity of sampling methods.

G Experimental Setup and Evaluation Protocols

In this section, we detail the physical deployment configurations used to evaluate the discrete latent planning (DLP) and local path following capabilities of Hydra.

G.1 Goal-Directed Planning Evaluation

To evaluate the computational efficiency and planning capabilities of our DLP formulation, we deploy Hydra across two distinct physical embodiments: a wheeled differential-drive robot (Clearpath

Jackal) and a quadrupedal robot (Boston Dynamics Spot). Given that unmapped, long-horizon planning remains a largely unsolved challenge in autonomous navigation, our experimental scope is intentionally bounded to close-range, complex local navigation tasks.

We define two primary evaluation scenarios:

- **Variable-Initialization Point Navigation:** The robot must plan and execute a collision-free trajectory to a Cartesian coordinate approximately $d = 8.0$ meters away. To evaluate robustness to initial state perturbations, the robot is initialized in 5 varying starting orientations, sampled uniformly from $\theta_{init} \in [-40°, +40°]$ relative to the goal vector. This is evaluated in both unobstructed spaces and partially occluded environments.
- **Constrained Corridor Maneuvering:** The robot must execute a high-curvature turn (approximately $90°$) within a narrow corridor, testing the model's ability to respect tight geometric constraints during planning.

A trial succeeds if the robot navigates within 0.5 meters of the goal without collision. We define trial success $\mathcal{S}$ for a trajectory $\tau = \{x_0, x_1, \ldots, x_T\}$ with respect to a goal position $x_{goal} \in \mathbb{R}^2$ as:

$$\mathcal{S}(\tau) = \mathbb{1}\left(\left(\min_{t\in[0,T]} ||x_t - x_{goal}||_2 \leq 0.5 \text{ m}\right) \wedge \left(\sum_{t=0}^{T} \mathcal{C}(x_t) = 0\right)\right)$$

where $\mathcal{C}(x_t)$ is a binary collision indicator function at time step t.

Compute Architecture Constraints: Due to the computation demand of NWM, on-board edge computation is currently bottlenecked by memory bandwidth and FLOP limits. Consequently, all physical inference is offloaded to a high-compute workstation equipped with an NVIDIA A5000 GPU (24GB VRAM).

G.2 Local Path Following Settings

To isolate and evaluate the robustness of the downstream local path following policy, we construct three specialized physical scenarios representing distinct environmental priors:

- **Cluttered Goal-Conditioned Traversal (Obstructed Goal):** The robot is tasked with navigating a narrow corridor seeded with static, unmapped obstacles. This setting evaluates the policy's capacity for on-the-fly obstacle avoidance and reactive re-planning while maintaining progress toward a line-of-sight goal.
- **Multi-Waypoint Navigation (Long Distance):** The robot must execute a long-horizon navigation task. This includes navigating multiple intersections within a building's corridor network, culminating in a room entry and a stop adjacent to a designated target. This evaluates the model's temporal consistency over extended rollouts.
- **Strict Boundary Adherence (Indoor Corridor):** The robot operates in a narrow, feature-sparse indoor corridor. The primary objective is to maintain a safe margin from the walls. This setting tests the model's geometric boundary perception in constrained space.

Performance is measured using strict geometric bounds (*e.g.*, L_2 norm to the goal) and boolean collision constraints.