Language-Conditioned World Modeling for Visual Navigation

Yifei Dong[1,*] **Fengyi Wu**[1,*] **Yilong Dai**[1] **Lingdong Kong**[1] **Guangyu Chen**[1] **Yetong Sha**[1]
Qiyu Hu[1] **Feng Liu**[2] **Siyu Huang**[3] **Qi Dai**[4] **Zhi-Qi Cheng**[1,†]

[1]University of Washington [2]Drexel University [3]Clemson University [4]Microsoft Research

Code: https://github.com/UWMILab/LCVN
Project Page: https://uwmilab.github.io/LCVN-webpage/
Dataset: https://huggingface.co/datasets/fly1113/LCVN

Abstract

Goal-conditioned visual navigation has been a long-standing testbed for embodied AI. We study a natural language-conditioned variant, **language-conditioned visual navigation (LCVN)**, in which an embodied agent must follow a natural language instruction given only an initial egocentric observation. Without access to goal images, the agent must rely on language to shape its perception and continuous control. We introduce the LCVN Dataset, a benchmark of 39,016 trajectories and 117,048 human-verified instructions spanning diverse environments and instruction styles. Building on this benchmark, we study two complementary paradigms: (i) latent-imagination policy learning, in which a diffusion-based world model (LCVN-WM) imagines future observations and an actor–critic agent (LCVN-AC) learns its policy entirely within the imagined latent space; and (ii) unified autoregressive prediction, in which a single multimodal backbone (LCVN-Uni) jointly predicts actions and observations in one forward pass over a shared token sequence. Experiments show that the two instantiations offer complementary strengths: latent imagination produces more temporally coherent rollouts, whereas the unified autoregressive agent, which benefits from a pre-trained multimodal backbone, transfers better to unseen environments. Targeted ablations further isolate the contributions of language guidance, conditioning signals, and instruction style, clarifying when language grounding versus dynamics modeling is the performance bottleneck. Together, these findings position LCVN as a concrete testbed for studying how language, imagination, and decision-making interact in embodied agents.

1 Introduction

Goal-conditioned visual navigation has emerged as a central testbed for embodied AI, where an agent predicts a sequence of actions toward a specified goal from egocentric visual observations [78, 74, 66, 11, 29, 27, 68]. A dominant line of work specifies the goal as a visual image, a snapshot of the destination, and trains policies to predict trajectories that reach it [74, 78]. These goal-conditioned policies are typically trained on offline datasets of recorded egocentric trajectories and evaluated by their ability to predict future actions from a single starting observation, a protocol that has enabled large-scale data sharing across robots and environments [73].

[*]Equal contribution, listed in random order.
[†]Corresponding author.

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

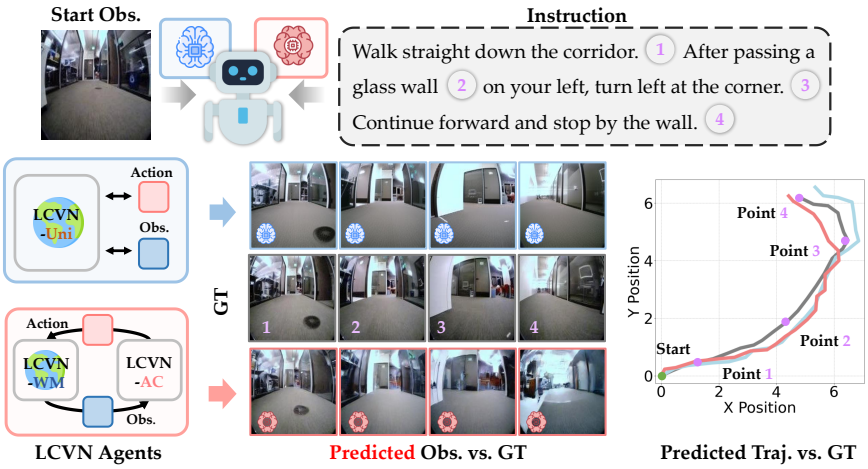

Figure 1: **Language-Conditioned Visual Navigation (LCVN).** Given only an initial egocentric observation and a language instruction, the agent generates the entire future trajectory without environmental feedback, imagining intermediate states (①–④) along the described route. **LCVN-WM + LCVN-AC** imagines future latent observations and selects actions within the imagined latent space, while **LCVN-Uni** jointly predicts the next action and observation in a single autoregressive pass, conditioned on both visual and language inputs.

Visual goals, however, can be a limiting interface for many real-world settings, since users rarely have a photograph of the intended destination at hand. A more natural and information-rich alternative is natural language, which can convey not only a target location but also intermediate landmarks, directional cues, and broader contextual constraints [97, 90]. A brief instruction such as *"After passing a glass wall on your left, turn left at the corner."* often suffices to guide action even in unfamiliar places, and humans seamlessly integrate such guidance with what they see. Building embodied agents that can do the same, grounding language effectively in perception and continuous control, remains a core open challenge [6, 47, 46, 61].

A central difficulty is tightly coupling perception, prediction, and decision-making [53, 54, 9, 60]. A promising route is to learn a world model as a neural simulator of environment dynamics, acquiring directly from data a predictive model of how observations evolve under actions [32, 94, 53, 100], which enables imagining future observations and planning over extended horizons. However, many existing designs [44, 7] focus on visual prediction while remaining decoupled from policy learning, limiting how agents adjust actions in response to linguistic guidance. Recent unified approaches [23] mitigate this but suffer from inefficient interleaved training and inference. These limitations call for methods that better align language grounding, neural simulation, and policy learning.

To study this problem, we introduce **language-conditioned visual navigation (LCVN)** task (Fig. 1), in which an agent receives a natural language instruction and a single egocentric starting observation, and predicts the entire future trajectory toward the destination it describes. Following the trajectory-prediction protocol of prior goal-conditioned navigation work [74, 78, 73], training and evaluation operate on offline datasets of recorded trajectories, providing a reproducible testbed in which different solution approaches can be compared on identical inputs without dependence on interactive simulators. To support rigorous and systematic evaluation, we further construct **LCVN Dataset**, a large-scale corpus of language-annotated trajectories spanning diverse environments and instruction styles.

Building on this benchmark, we instantiate and systematically compare two paradigms that couple language grounding, future-state prediction, and policy learning in qualitatively different ways. The first paradigm (Fig. 2), latent-imagination policy learning, trains a language-conditioned diffusion world model **LCVN-WM** that imagines future latent observations, paired with an actor–critic agent **LCVN-AC** that learns its policy entirely within the imagined latent space via intrinsic rewards. The second paradigm (Fig. 3), unified autoregressive prediction, fine-tunes a language-conditioned autoregressive world model **LCVN-Uni** that jointly predicts actions and observations in a single forward pass over a shared token sequence.

Extensive experiments show that LCVN agents outperform strong baselines on both navigation and imagination benchmarks, with latent-imagination policy learning producing more temporally

coherent rollouts and unified autoregressive prediction generalizing better to unseen environments. Targeted ablations on language guidance, conditioning signals, and instruction style surface three findings that reframe what language-conditioned navigation actually requires: (i) language guidance functions as the goal specification that aligns states with destinations, supporting both navigation success and long-horizon rollout stability; (ii) primary bottleneck lies in grounding linguistic intent into continuous control rather than in language understanding itself; and (iii) navigation-relevant signal density matters more than descriptive richness. In summary, our contributions are:

1. We formalize **language-conditioned visual navigation** task as a natural language variant of goal-conditioned navigation, and release the **LCVN Dataset**, a large-scale collection of 39,016 trajectories and 117,048 human-verified instructions in three styles (concise, intricate, landmark-based), assembled from five heterogeneous robotic platforms.
2. We instantiate and systematically compare two paradigms on LCVN task: latent-imagination policy learning, in which a diffusion-based world model **LCVN-WM** imagines future visual states and an actor–critic agent **LCVN-AC** learns its policy entirely within the imagined latent space; and unified autoregressive prediction, in which a single multimodal backbone **LCVN-Uni** jointly predicts actions and observations in one forward pass.
3. We provide experiments and ablations that characterize the behavior and generalization of both paradigms on the LCVN task, positioning LCVN as a shared testbed for studying how language, imagination, and decision making shape embodied behavior.

2 Related Work

Visual Navigation spans goal-conditioned and free-exploration settings, with goal-conditioned navigation long regarded as a core robotics challenge [78, 74, 66, 11, 29, 27, 68]. In this setting, an agent receives context and target images and must generate trajectories to reach the goal [74, 78, 20]. Early systems combined neural SLAM with analytical planning [11], while later work learned direct mappings from egocentric observations to actions with reinforcement learning [13, 55, 33]. More recent foundation-model approaches [74, 78, 73] train policies on offline datasets of recorded trajectories to predict future actions from a single starting observation and a goal specification, a protocol that has enabled large-scale data sharing across heterogeneous robots and environments. However, these approaches typically specify the goal as a visual image, underutilizing the intent and contextual cues that language can convey [97, 90]. In parallel, language-guided navigation has been studied in ObjectNav [93, 21], VLN [3, 49], and REVERIE [70], which evaluate agents in simulators where observations are received at every step. LCVN occupies a distinct point: unlike discrete graphs [3, 70] or simulator-bound continuous control [49, 93], it inherits the offline trajectory-prediction protocol of goal-conditioned navigation with language goals over recorded real-robot trajectories, complementing closed-loop evaluation and scaling to heterogeneous real-world data across diverse platforms and environments.

World Models in Navigation. World models can be understood as learned neural simulators of environment dynamics [22, 100, 17]: rather than relying on hand-built physics or rendering engines, they acquire a predictive model of how observations evolve under actions directly from data. This has driven a rapid evolution from recurrent latent dynamics [32, 34, 35, 36] to Transformer-based [4, 8, 42, 5] and diffusion-based architectures [10, 1, 86, 7, 95], with LLM-driven variants further modeling dynamics via language prompting [99, 92, 88]. Within navigation, two design philosophies have emerged. Policy-centric methods [73, 74, 78] map observations directly to actions, treating future-state prediction as implicit. Model-centric methods explicitly predict future observations for planning [59, 16, 15], instantiated as GAN-based imagination in PathDreamer [44], diffusion video rollouts in NWM [7], and unified autoregressive prediction in UniWM [23]. Yet these works largely decouple imagination from policy learning: NWM omits language entirely, PathDreamer's auxiliary inputs hinder generalization [56], and UniWM's interleaved training and inference are inefficient. How language grounding, neural simulation, and policy learning should be jointly designed under a single task remains underexplored. LCVN provides such a setting, where we instantiate two paradigms: latent-imagination policy learning, pairing a diffusion-based world model LCVN-WM with an actor–critic agent LCVN-AC trained in its latent space; and unified autoregressive prediction, where a single multimodal backbone LCVN-Uni jointly predicts actions and observations.

3 LCVN: Language-Conditioned Visual Navigation

In this section, we formalize the **language-conditioned visual navigation** task, a natural-language variant of goal-conditioned visual navigation. An embodied agent begins navigation with an egocentric RGB observation o_s and receives a natural language instruction $I = \langle w_1, w_2, \ldots, w_n \rangle$, consisting of a sequence of words. The agent's goal is to generate a sequence of navigation actions $A_T = \{\hat{a}_1, \hat{a}_2, \ldots, \hat{a}_T\}$ that successfully guides it to the destination described by the instruction. Each action $\hat{a}_t$ can take the form of a continuous control command $(\mathbf{u}_t, \phi_t)$. In detail, the planar displacement $\mathbf{u}_t \in \mathbb{R}^2$ represents forward, backward, and lateral motion, together with a yaw rotation $\phi_t \in \mathbb{R}$. The action sequence terminates with a null action indicating arrival at the goal location. Following the trajectory-prediction protocol of prior goal-conditioned navigation work [74, 78, 73], the agent predicts this entire action sequence from the initial observation o_s and the instruction I, and is evaluated on offline datasets of recorded trajectories by comparing the predicted trajectory against the recorded ground truth. The instruction I serves as a persistent contextual condition that modulates the agent's policy throughout the episode.

3.1 LCVN Dataset

To support research in language-conditioned visual navigation, we construct **LCVN**, a large-scale multi-style instruction dataset built by augmenting five heterogeneous robotic trajectory datasets with human-verified language instructions. We use **Go Stanford** [37], **ReCon** [72], **SCAND** [41], and **HuRoN** [38] for training and in-domain evaluation, covering complementary aspects of real-world navigation (e.g., open-world settings in ReCon, socially compliant behavior in SCAND), and reserve **TartanDrive** [85] as an unseen split for zero-shot evaluation, since its off-road driving scenes differ markedly from the indoor and structured outdoor environments seen in training, providing a strong distribution shift for testing generalization. Trajectories are standardized by normalizing per-frame displacement by each dataset's average step size, filtering out backward movements and trajectories shorter than three steps [7, 78], and segmented into semantically coherent sub-scenes using Qwen-3.5-9B [83]. We adopt a seeded few-shot annotation strategy: our annotation team first manually writes a seed set of high-quality instructions, which then serves as the few-shot prompt context for Qwen-3.5-9B to generate instructions at scale [62], followed by human verification and refinement. Alongside the egocentric video, we additionally render a top-down 2D trajectory map as visual input, which disambiguates global heading changes from local motion noise. Each trajectory receives instructions in three styles: **Concise** (essential directional cues), **Intricate** (rich scene descriptions), and **Landmark-based** (anchored to environmental landmarks), enabling systematic evaluation of language grounding under varying levels of detail rather than a single uniform style [45].

The final LCVN dataset contains 39,016 trajectories and 117,048 instructions, partitioned into training (28,813), val seen (3,602), val unseen (1,500, solely from TartanDrive), and test (5,101, from both TartanDrive and other sources). Beyond its primary application in language-conditioned visual navigation, LCVN dataset may also support related tasks like style-aware instruction generation [48, 90] and multimodal understanding [84, 75, 50, 96]. To verify annotation quality, a human evaluation on 640 randomly sampled trajectories yields an overall quality score of 5.2/6.0, confirming instruction quality across all three styles. Please kindly refer to **Appendix** for more dataset details.

4 Paradigms for Language-Conditioned Visual Navigation

To solve the LCVN task, we instantiate and compare two paradigms that couple language grounding, future-state prediction, and policy learning in qualitatively different ways, under identical training data to enable a controlled comparison. Paradigm 1: Latent-imagination policy learning (Sec. 4.1–4.2, Fig. 2) trains a language-conditioned diffusion world model **LCVN-WM** that imagines future latent observations, paired with an actor–critic agent **LCVN-AC** that learns its policy within this latent space via intrinsic rollout rewards. Paradigm 2: Unified autoregressive prediction (Sec. 4.3, Fig. 3) fine-tunes a language-conditioned autoregressive world model **LCVN-Uni** that jointly predicts the next action and observation in a single forward pass over a shared token sequence.

4.1 LCVN-WM

LCVN-WM is the imagination component of Paradigm 1: it predicts future latent observations conditioned on past observations, the current action, and the language instruction, providing the

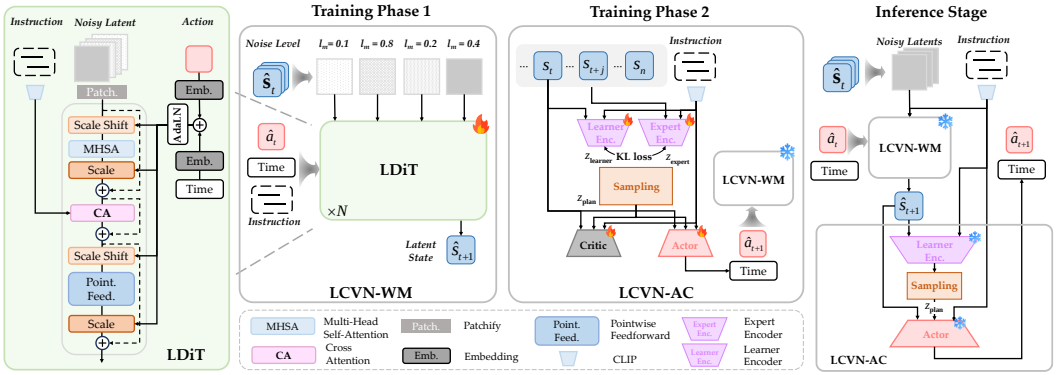

Figure 2: **Language-conditioned world model (LCVN-WM) and associated LCVN-AC. Phase 1:** LCVN-WM is trained with Diffusion Forcing [12, 77] to predict future latent observations from noisy context latents at independent noise levels, conditioned on actions, instruction, and time shift. **Phase 2:** LCVN-AC is trained in LCVN-WM's latent space, aligning expert and learner plans via KL divergence and optimizing an actor–critic objective with intrinsic rewards that measure agreement between predicted and expert latent rollouts. **Inference:** LCVN-WM predicts the next latent observation, on which LCVN-AC conditions to generate the next action.

imagined trajectories on which LCVN-AC's policy is trained. Architecturally, LCVN-WM extends the Diffusion Transformer (DiT) [69] with language conditioning (termed LDiT) and employs diffusion forcing (DF) [12, 77] to enable flexible noising patterns, illustrated in Fig. 2.

Given current observations encoded as VAE latents, LCVN-WM F_θ conditions on navigation action $\hat{a}_t$ where t denotes timestep, instruction I, and time shift t_s to predict the next state $\hat{s}_{t+1}$:

$$s_i = \text{Enc}_\theta(o_i) \quad \hat{s}_{t+1} \sim F_\theta\big(\hat{s}_{t+1} \mid \hat{\mathbf{s}}_t,\, \hat{a}_t, I, t_s\big), \tag{1}$$

where o_i denotes image observation and is encoded by a learned VAE encoder to form s_i, $\hat{\mathbf{s}}_t = [\hat{s}_t, \hat{s}_{t-1}, ..., \hat{s}_{t-k}]$ represents the previous estimate of the state, and k is the context size.

Language Conditioning. To enable language conditioning, we encode the instruction I into I_{clip} using a frozen CLIP text encoder [71]. The LDiT architecture integrates a multi-head cross-attention layer within each DiT block, positioned between the self-attention and feed-forward layers, as shown on the right side of Fig. 2. This design enables flexible interaction between visual latents and the instruction embeddings I_{clip}, which are directly fed into the cross-attention mechanism, thereby promoting semantic consistency between states and goals.

Meanwhile, we also include action and time conditioning within the conditioning mechanism following the settings in [7]. The navigation action $\hat{a}_t$ is encoded by first applying sine–cosine positional encoding to each scalar, followed by a MLP, and then concatenating results into a single vector. A similar procedure is applied to encode relative timeshift t_s and diffusion timestep. The final vector is obtained by summing all three embeddings, which is then used in an AdaLN layer to generate scale and shift parameters that modulate outputs of Layer Normalization and intermediate activations in each attention block.

Diffusion Forcing. LCVN-WM adopts the DF paradigm to strengthen temporal modeling in long-horizon navigation. Unlike standard diffusion training, DF applies independent noise levels ℓ_m to each latent state $\hat{s}_{t-m}$ within the context sequence $\hat{\mathbf{s}}_t = [\hat{s}_t, \hat{s}_{t-1}, \ldots, \hat{s}_{t-k}]$:

$$\hat{s}_{t-m}^{(\ell_m)} = \sqrt{\alpha_{\ell_m}}\, \hat{s}_{t-m} + \sqrt{1 - \alpha_{\ell_m}}\, \epsilon_m, \tag{2}$$

where $\epsilon_m \sim \mathcal{N}(0, I)$ and α_{ℓ_m} follow the standard noise schedule. DF introduces heterogeneous corruption across the context window, encouraging stronger temporal modeling [12, 77].

4.2 LCVN-AC

LCVN-AC is the policy component of Paradigm 1: while LCVN-WM imagines latent rollouts, LCVN-AC learns a language-conditioned policy that selects actions to follow the instruction, trained entirely within LCVN-WM's latent space (Fig. 2). Concretely, the agent learns a policy $\pi_\theta(\hat{a}_{t+1} \mid s_t, I_{clip})$ and a value function $v_\psi(s_t, I_{clip})$ from the LCVN dataset, where π_θ denotes the actor, v_ψ the critic, and I_{clip} the CLIP-encoded instruction embedding.

Trajectory Encoding and Language Conditioning. Following [65, 67], we replace high-dimensional RGB images with compact world-model latents. In addition to encoding first-person observations, we represent a latent trajectory segment from the current state to the goal, consisting of continuous navigation actions $a \in \mathbb{R}^3$ across timesteps, and compress each trajectory into a latent plan z_{plan} using a seq2seq CVAE [64, 91]. During training, an expert encoder q_{expert} consumes the full trajectory to produce expert plan z_{expert}, while a learner encoder q_{learner} maps (s_t, I_{clip}) at test time to predict z_{plan}. We align these encoders by minimizing the KL divergence:

$$\mathcal{L}_{\text{KL}} = D_{\text{KL}}\Big(q_{\text{expert}}(z_{\text{expert}} \mid s_{t:n}, I_{clip}) \,\|\, q_{\text{learner}}(z_{\text{plan}} \mid s_t, I_{clip})\Big), \tag{3}$$

where n denotes total trajectory length. Aligning expert and learner encoders with instruction embeddings improves efficiency and enforces semantic consistency between states and goals, ensuring latent plans remain grounded in the intended destination. For language conditioning, we maximize cosine similarity between LCVN-WM latents and corresponding instruction embeddings while minimizing it for mismatched pairs [71, 67, 81], resulting in loss $\mathcal{L}_{\text{ins}}$.

Actor–Critic with Intrinsic Reward. At each step, we sample a latent plan $z_{\text{plan}} \sim q_{\text{learner}}(\cdot \mid s_t, I_{clip})$, then use actor π_θ to generate action $\hat{a}_t \sim \pi_\theta(\cdot \mid s_t, I_{clip}, z_{\text{plan}})$. The next latent state is predicted by frozen LCVN-WM F_θ:

$$\hat{s}^\pi_{t+1} \sim F_\theta(\hat{s}^\pi_{t+1} \mid \mathbf{s}_t, \hat{a}_t, I_{clip}, t_s), \tag{4}$$

where $\mathbf{s}_t = [s_t, s_{t-1}, ..., s_{t-k}]$ denotes the sequence of previous latent states and k is the context size. For the initial step ($t = 0$), when past states $s_{t-1}, \ldots, s_{t-k}$ are unavailable, we pad the history by repeating the initial latent s_s. To mitigate covariate shift, we compute an intrinsic reward [18] that measures the alignment between predicted and expert trajectories in latent space:

$$r^{\text{int}}_t(s^E_{t+1}, \hat{s}^\pi_{t+1}) = \frac{s^E_{t+1} \cdot \hat{s}^\pi_{t+1}}{\max(\|s^E_{t+1}\|, \|\hat{s}^\pi_{t+1}\|)^2}\,, \tag{5}$$

where s^E_{t+1} is expert latent state at step $t+1$. The critic v_ψ estimates expected discounted return by:

$$v_\psi(s_t, I_{clip}, z_{\text{plan}}) \approx \mathbb{E}_{\pi_\theta, p_\phi}\left[\sum\nolimits_{k=t}^{t+H-1} \gamma^{k-t} r^{\text{int}}_k\right], \tag{6}$$

where $\gamma \in [0, 1]$ is the discount factor and H is the rollout horizon. This formulation allows the critic to estimate the cumulative intrinsic reward under the current policy π_θ and latent plan z_{plan}. We repeat this rollout over the entire trajectory horizon to produce $\{\hat{s}^\pi_k\}^n_{k=t+1}$, and compute intrinsic rewards and discounted returns for critic supervision.

During training, the critic regresses to λ-return [35]:

$$\mathcal{L}(\psi) = \mathbb{E}_{\pi_\theta, p_\phi}\left[\sum\nolimits_{k=t}^{t+H-1} \frac{1}{2}\left(v_\psi(\hat{s}_k, I_{clip}, z_{plan}) - \text{sg}(V^\lambda_k)\right)^2\right], \tag{7}$$

where $\text{sg}(\cdot)$ is the stop-gradient operator, p_ϕ denotes transition dynamics of LCVN-WM, and V^λ_k represents λ -return computed with a slowly updated target critic v_ψ. On the other hand, the actor maximizes expected return while regularizing plan consistency and language alignment:

$$\mathcal{L}(\theta) = \mathbb{E}_{\pi_\theta, p_\phi}\left[\sum\nolimits_{k=t}^{t+H-1}\left(-V^\lambda_k + \alpha_1 \mathcal{L}^k_{\text{KL}} + \alpha_2 \mathcal{L}^k_{\text{ins}}\right)\right]. \tag{8}$$

At inference time, agent plans next action based on current latent state $\hat{s}_{t+1}$ predicted by LCVN-WM, together with instruction embedding I_{clip} and z_{plan}:

$$\hat{a}_{t+1} \sim \pi_\theta(\hat{a}_{t+1} \mid s_{t+1}, I_{clip}, z_{\text{plan}})\,. \tag{9}$$

4.3 LCVN-Uni

LCVN-Uni instantiates Paradigm 2: instead of separating an explicit world model from a downstream policy, it folds imagination and planning into a single multimodal autoregressive backbone G_θ (Fig. 3) that produces the next action and the next observation in one forward pass:

$$(\hat{a}_{t+1},\, \hat{o}_{t+1}) \;=\; G_\theta\big(\hat{o}_t,\, \hat{a}_t,\, I,\, o_s\big). \tag{10}$$

Architecturally, LCVN-Uni builds on [23, 82], but trains action and observation prediction jointly rather than alternately, sharing representations between planning and imagination at every step.

Data Preprocessing. Each training sample consists of (o_t, a_t, o_s, I) paired with two targets, the next action $\hat{a}_{t+1}$ and the next observation $\hat{o}_{t+1}$ (see **Appendix** for prompt design). This eliminates need for separate sample types for planning and world modeling, and streamlines the training pipeline.

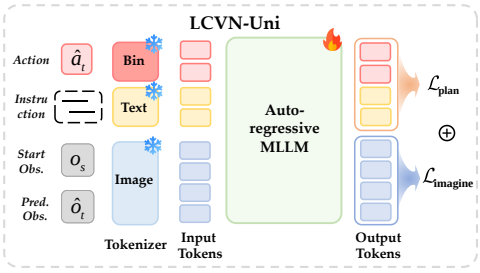

Figure 3: **LCVN-Uni architecture.** LCVN-Uni unifies navigation planning and world modeling within an autoregressive MLLM backbone. Actions $\hat{a}_t$, instructions I, and observations $o_s, \hat{o}_t$ are tokenized by bin, BPE, and VQ tokenizers, respectively, then fused into a unified sequence.

Multimodal Tokenization. We employ three tokenizers [23] to couple multimodal inputs: a VQ image tokenizer [30] for visual observations o_t and o_s, a BPE tokenizer [82] for instructions I and text prompts, and a bin tokenizer [23] for actions a_t. The resulting token sequences are fed to a causal Transformer that jointly models planning and imagination.

Joint Training Objective. At each training step, the model produces logits for both action bins and visual tokens, enabling end-to-end joint optimization:

$$\mathcal{L}_{\text{joint}} = \mathcal{L}_{\text{plan}} + \lambda \mathcal{L}_{\text{imagine}}, \tag{11}$$

where $\mathcal{L}_{\text{plan}}$ is the discretized bin token loss [23] for action prediction, $\mathcal{L}_{\text{imagine}}$ [23, 51] is the reconstruction loss for predicted future observations, and λ balances the two. This shared objective lets the same backbone learn representations that serve both planning and visual prediction.

Inference. At test time, LCVN-Uni generates the next action and observation in a single forward pass, avoiding alternating substeps used in [23]. The predicted observation $\hat{o}_{t+1}$ is fed back as input observation at next step, so the agent unrolls its trajectory entirely in its own imagined visual stream.

5 Experiments

5.1 Experimental Settings

Baselines. We adopt three navigation baselines that leverage SOTA world models: Diamond [2] (UNet-based diffusion), NWM [7] (CDiT-based diffusion), and NWM (lang), our language-augmented variant of NWM where CLIP-encoded instructions are summed with action and time embeddings and fed through CDiT's AdaLN block. We further include two policy-only baselines, GNM [73] (lang) and NoMaD [78] (lang), adapted by replacing their image-goal encoder with a frozen CLIP text encoder followed by an MLP that projects instructions into the original goal-token space. Each baseline is retrained from scratch on the LCVN training split, with LCVN-AC trained specifically for each world-model baseline. To isolate language guidance, we evaluate two ablated variants: LCVN-Uni (w/o ins), which removes instructions only from observation-prediction substep, and LCVN-WM (w/o ins), trained without language conditioning.

Evaluation Metrics. We evaluate performance using two suites of metrics. **(1) Navigation quality.** We report Absolute Trajectory Error (ATE), Relative Pose Error (RPE) [79], and Success Rate (SR). A trajectory is considered successful if the final distance to goal is smaller than agent's average step size (in meters). **(2) Imagination quality.** To assess fidelity of navigation visualizations, we adopt structural and perceptual metrics including SSIM [89], PSNR [39], LPIPS [98], and DreamSim [28]. To further evaluate long-horizon stability under rollout, we report SSIM@n, PSNR@n, LPIPS@n, and DreamSim@n. Due to space limits, kindly refer to **Appendix** for details.

Datasets & Implementation Details. Experiments are conducted on our proposed LCVN dataset (Sec. 3.1) using the val seen, val unseen, and test splits. For LCVN-WM and LCVN-AC, we train an ImageVAE [43] from scratch to compress images into 32×32 latents. LCVN-Uni is fine-tuned on GAIR Anole-7B [14] (4096-token context), while keeping the text and image tokenizers as well as the bin-token encoder frozen. Their comparison evaluates practical instantiations. Input images are resized to 448×448 (height $\times$ width) and discretized into 784 visual tokens when context size is 1 or 2, and into 625 tokens when context size is 4. All models are trained on 4$\times$NVIDIA A100 GPUs (80GB each). Due to space limits, kindly refer to **Appendix** for additional implementation details.

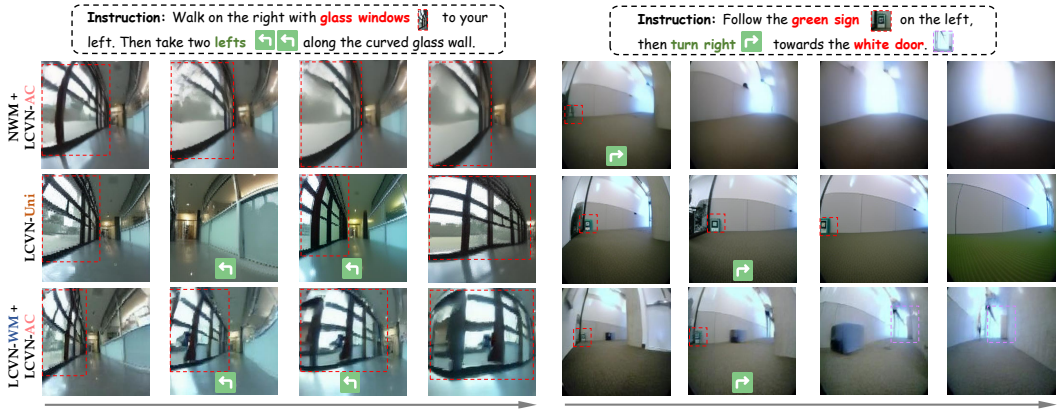

Figure 4: **Qualitative Comparisons** on LCVN val seen split across NWM and LCVN agents. LCVN agents exhibit stronger sensitivity to directional changes and are less prone to losing landmarks during world modeling.

Table 1: **Comparison with SOTA Methods** upon Language-Conditioned Visual Navigation on LCVN val seen, val unseen and test splits with SR, ATE, and RPE. We equip two LCVN agents with context sizes k set to 1, 2, 4.

Methods	Context Size	Validation Seen			Validation Unseen			Test		
		ATE↓	RPE↓	SR↑	ATE↓	RPE↓	SR↑	ATE↓	RPE↓	SR↑
GNM [73] (lang)	4	1.18	0.39	0.21	2.72	0.91	0.10	1.89	0.55	0.16
NoMaD [78] (lang)	4	1.08	0.36	0.24	2.61	0.88	0.11	1.78	0.53	0.18
Diamond [2] + LCVN-AC	4	1.35	0.42	0.18	2.84	0.95	0.09	2.05	0.58	0.14
NWM [7] + LCVN-AC	4	0.75	0.24	0.28	2.17	0.73	0.12	1.33	0.42	0.21
NWM [7] (lang) + LCVN-AC	4	0.72	0.26	0.29	2.21	0.70	0.13	1.31	0.44	0.22
LCVN-Uni	1	0.43	0.13	0.37	1.36	0.52	0.21	0.74	0.25	0.32
LCVN-Uni (w/o ins)	2	0.47	0.15	0.34	1.49	0.58	0.19	0.81	0.28	0.29
LCVN-Uni	**2**	0.36	**0.11**	0.42	**1.21**	0.47	**0.27**	**0.65**	0.22	**0.36**
LCVN-Uni	4	0.37	0.12	0.41	1.24	**0.43**	0.25	**0.65**	**0.21**	0.35
LCVN-WM + LCVN-AC	1	0.39	0.15	0.39	1.62	0.61	0.18	0.81	0.29	0.31
LCVN-WM + LCVN-AC	2	0.36	0.13	0.42	1.55	0.59	0.19	0.78	0.27	0.33
LCVN-WM (w/o ins) + LCVN-AC	4	0.46	0.16	0.35	1.82	0.67	0.15	0.89	0.31	0.29
LCVN-WM + LCVN-AC	**4**	**0.34**	0.12	**0.43**	1.51	0.56	0.19	0.76	0.26	0.34

Table 2: **Comparison with SOTA methods** on imagination performance (single-frame / long-horizon) on LCVN val seen split. We equip two LCVN agents with context sizes k set to 1, 2, 4. Best results are in bold.

Methods	Context Size	Single-frame Generation				Long-horizon Generation @8			
		SSIM ↑	PSNR ↑	LPIPS ↓	DreamSIM ↓	SSIM@8 ↑	PSNR@8 ↑	LPIPS@8 ↓	DreamSIM@8 ↓
Diamond [2]	4	0.315	9.850	0.427	0.135	0.114	4.526	0.679	0.283
NWM [7]	4	0.370	11.425	0.314	0.096	0.181	6.057	0.629	0.196
NWM [7] (lang)	4	0.382	11.612	0.309	0.094	0.176	6.041	0.605	0.191
LCVN-Uni	1	0.398	12.881	0.306	0.076	0.201	7.057	0.466	0.128
LCVN-Uni (w/o ins)	2	0.387	12.642	0.319	0.082	0.192	6.874	0.508	0.135
LCVN-Uni	**2**	0.423	13.325	0.298	**0.072**	0.218	7.412	0.451	**0.119**
LCVN-Uni	4	0.421	13.301	0.293	0.073	0.216	7.395	0.445	0.121
LCVN-WM	1	0.419	18.895	0.210	0.087	0.267	10.530	0.336	0.134
LCVN-WM	2	0.427	19.742	0.198	0.081	0.281	11.084	0.322	0.131
LCVN-WM (w/o ins)	4	0.395	17.247	0.236	0.118	0.241	9.580	0.371	0.161
LCVN-WM	**4**	**0.435**	**20.316**	**0.189**	0.078	**0.293**	**11.527**	**0.315**	0.127

5.2 Comparisons to SOTA Methods

Navigation performance. Table 1 reveals a pattern aligned with structural differences between the two paradigms (qualitative results in Fig. 4). LCVN-WM + LCVN-AC achieves the strongest in-domain performance via high-fidelity LDiT rollouts, but this advantage shrinks on val unseen as rollouts may become unreliable on novel scenes. LCVN-Uni shows the opposite profile, leveraging Anole-7B's pre-trained multimodal prior and avoiding the two-stage error-compounding bottleneck. Absolute SR remains moderate across all methods, reflecting the inherent difficulty of LCVN: predicting entire trajectories without corrective feedback causes errors to compound over the horizon. Despite this, LCVN agents substantially surpass other baselines, establishing a strong starting point and validating both paradigms as promising directions.

Table 3: Ablations of **language, action and time conditioning** on navigation and imagination performance of LCVN-WM (k=4) on LCVN val seen split. ✓ denotes with and × denotes without. Best results are in bold.

			Navigation			Imagination			
Language	**Action**	**Time**	**ATE↓**	**RPE↓**	**SR↑**	**SSIM↑**	**DreamSIM↓**	**SSIM@8↑**	**DreamSIM@8↓**
×	×	✓	1.82	0.63	0.12	0.251	0.627	0.095	0.798
✓	×	×	1.12	0.35	0.19	0.341	0.137	0.187	0.226
×	✓	×	0.54	0.22	0.31	0.388	0.112	0.229	0.169
✓	×	✓	0.89	0.31	0.22	0.352	0.142	0.201	0.214
✓	✓	×	0.37	0.14	0.41	0.422	0.081	0.275	0.138
×	✓	✓	0.46	0.16	0.35	0.395	0.118	0.241	0.161
✓	✓	✓	**0.34**	**0.12**	**0.43**	**0.435**	**0.078**	**0.293**	**0.127**

Table 4: Ablations of **instruction style** for LCVN-Uni (k=2) and LCVN-WM (k=4) on LCVN val seen split.

		Navigation			Imagination			
Method	**Ins. Style**	**ATE↓**	**RPE↓**	**SR↑**	**SSIM↑**	**DreamSIM↓**	**SSIM@8↑**	**DreamSIM@8↓**
LCVN-Uni	Concise	0.37	0.13	0.41	0.421	0.076	0.218	0.121
	Intricate	0.39	0.11	0.38	0.415	0.074	0.211	0.123
	Landmark.	**0.32**	**0.10**	**0.47**	**0.433**	**0.069**	**0.225**	**0.116**
LCVN-WM	Concise	0.35	0.12	0.42	0.435	0.078	0.293	0.127
	Intricate	0.36	0.14	0.40	0.429	0.081	0.285	0.128
	Landmark.	**0.31**	**0.11**	**0.47**	**0.442**	**0.076**	**0.302**	**0.125**

Imagination performance. Table 2 compares imagination abilities of LCVN-Uni and LCVN-WM against SOTA baselines. Both LCVN agents achieve competitive results across single-frame and long-horizon evaluations. On one-step predictions, LCVN-WM ($k = 4$) attains the highest structural fidelity, while LCVN-Uni ($k = 2$) achieves the best perceptual alignment. Under multi-step rollouts, both agents maintain stability and mitigate compounding errors over longer horizons.

Failure Case Analysis. A representative failure case appears in Fig. 4 (right): LCVN-Uni predicts the correct turning direction but only imagines a few post-turn view transitions: the corridor and target white door are not correctly synthesized. This reflects its weaker long-horizon imagination (Table 2), causing subsequent actions to lose visual grounding despite correct directional intent.

5.3 Ablation Studies

1) Language Guidance. We ablate the role of language guidance by comparing LCVN agents trained with and without instructions in Tables. 1, 2, and Fig. 5. Removing language consistently weakens both navigation and imagination: LCVN agents without instruction inputs exhibit lower success rates and less stable rollouts. In this task, language instructions serve as the goal specification, providing semantic alignment between states and destinations and thereby guiding effective navigation.

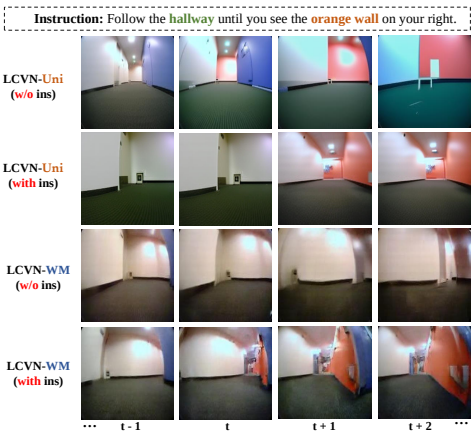

Figure 5: Qualitative comparisons of language guidance showcase both baselines better preserve semantic consistency across states with language.

2) Context Size. Tables 1 and 2 show that larger contexts generally improve both LCVN methods. LCVN-Uni peaks at $k = 2$ because Anole-7B's fixed 4096-token window forces a trade-off: images use 784 tokens at k = 1,2 but only 625 at $k = 4$, and the lower resolution offsets the longer horizon. We therefore adopt $k = 2$ for LCVN-Uni and $k = 4$ for LCVN-WM + LCVN-AC.

3) Language, Action and Time Conditioning. We train LCVN-WM with joint conditioning on language, action, and time, and ablate each in Table 3. Time conditioning alone is insufficient for meaningful navigation. Adding action conditioning on top of time yields a substantial improvement, as explicit control signals are essential for grounding latent dynamics. Further adding language

Table 5: **Pixel vs. Latent space** on the performance of LCVN-WM (k=4) on LCVN val seen split.

	Navigation			Imagination			
Encoding Space	**ATE↓**	**RPE↓**	**SR↑**	**SSIM↑**	**DreamSIM↓**	**SSIM@8↑**	**DreamSIM@8↓**
Pixel	0.42	0.16	0.36	0.381	0.095	0.257	0.150
Latent	**0.34**	**0.12**	**0.43**	**0.435**	**0.078**	**0.293**	**0.127**

Table 6: **External data scaling** on imagination performance of NWM and LCVN-WM (k=4) on LCVN val unseen. ✓ denotes training with additional Ego4D data. Best results are in bold.

Method	**External Data**	**SSIM↑**	**DreamSIM↓**	**SSIM@8↑**	**DreamSIM@8↓**
NWM	×	0.180	0.325	0.102	0.543
NWM	✓ (Ego4D)	0.197	0.318	0.115	0.528
LCVN-WM	×	0.302	0.239	0.167	0.376
LCVN-WM	**✓ (Ego4D)**	**0.313**	**0.226**	**0.184**	**0.357**

conditioning produces an additional gain, with language providing semantic alignment between states and goals to disambiguate where action-driven dynamics should lead. This progression indicates that primary bottleneck lies in grounding linguistic intent into continuous control.

4) Instruction Style. Table 4 shows that landmark-grounded instructions consistently yield the highest navigation success rates for both LCVN baselines, while concise instructions also outperform intricate ones in success rate. This pattern suggests that navigation-relevant cues may be more beneficial than descriptive richness alone for instruction following.

5) Latent *vs.* Pixel Space. Table 5 shows that training LCVN-WM in pixel space (PatchEmbed [25] following [77]) consistently underperforms its latent counterpart. Latent encoding also provides a compact shared representation [80] that integrates naturally with LCVN-AC under Paradigm 1.

6) External Data Scaling. Following the original NWM protocol, we train both NWM and LCVN-WM with additional Ego4D [31] data. Table 6 confirms that external data benefits generalization, with LCVN-WM maintaining a consistent advantage over NWM under both settings.

7) Intrinsic Reward. We evaluate policy-training variants for LCVN-AC ($k = 4$) on val seen with the world model frozen. Behavior cloning achieves 0.30 SR, increasing to 0.35 with plan-consistency and language-alignment losses. Full actor–critic training achieves 0.43 SR, while replacing the intrinsic reward with negative L2 distance yields 0.42, supporting the benefit of rollout-alignment training.

8) Initial Vision and World Modeling. We ablate initial visual input and world modeling with the same LCVN-Uni backbone. Both ablated variants remove the observation-prediction objective and imagined-frame feedback, while retaining the same training data and LoRA recipe. On val seen, the language-only variant achieves 0.11 SR. Adding the first frame raises SR to 0.24, versus 0.42 for full model. These results support benefits of initial vision and world modeling beyond language alone.

6 Conclusion

We introduced language-conditioned visual navigation (LCVN), a natural language variant of goal-conditioned visual navigation, supported by a large-scale LCVN dataset, and instantiated two paradigms: latent-imagination policy learning (LCVN-WM + LCVN-AC), which imagines futures with a diffusion world model and learns a latent-space actor–critic, and unified autoregressive prediction (LCVN-Uni), which jointly predicts actions and observations. Experiments reveal complementary strengths of the two instantiations: LCVN-WM + LCVN-AC produce more temporally coherent rollouts, while LCVN-Uni, built on a pre-trained multimodal backbone, achieves stronger navigation performance in unseen environments. Ablations further surface three findings: language guidance acts as the goal specification that aligns states with destinations and stabilizes long-horizon rollouts; the primary bottleneck lies in grounding linguistic intent into continuous control; and navigation-relevant signal density matters more than descriptive richness. We hope LCVN serves as a testbed for studying how language, imagination, and decision-making interact in embodied agents.

Acknowledgments

We acknowledge support from University of Washington faculty startup funding and USDOT through PacTrans's seed funding and small research projects programs; faculty support from the UW Undergraduate Research Faculty Fellows program (Henry Luce Foundation) and Engineering Unleashed/KEEN (The Kern Family Foundation); and student support from the Vicky L. Carwein and William B. Andrews Endowment for Graduate Programs and UW Graduate School Top Scholar and Conference Presentation Awards. We thank the reviewers for their feedback.

References

[1] Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, et al. Cosmos world foundation model platform for physical AI. *arXiv preprint arXiv:2501.03575*, 2025.

[2] Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos J Storkey, Tim Pearce, and François Fleuret. Diffusion for world modeling: Visual details matter in Atari. *Advances in Neural Information Processing Systems*, 37:58757–58791, 2024.

[3] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3674–3683, 2018.

[4] 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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15619–15629, 2023.

[5] Federico Baldassarre, Marc Szafraniec, Basile Terver, Vasil Khalidov, Francisco Massa, Yann LeCun, Patrick Labatut, Maximilian Seitzer, and Piotr Bojanowski. Back to the features: DINO as a foundation for video world models. *arXiv preprint arXiv:2507.19468*, 2025.

[6] Philip J. Ball, Jakob Bauer, Frank Belletti, et al. Genie 3: A new frontier for world models. https://deepmind.google/discover/blog/genie-3-a-new-frontier-for-world-models/, 2025.

[7] Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15791–15801, 2025.

[8] Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *arXiv preprint arXiv:2404.08471*, 2024.

[9] Hengwei Bian, Lingdong Kong, Haozhe Xie, Liang Pan, Yu Qiao, and Ziwei Liu. DynamicCity: Large-scale 4D occupancy generation from dynamic scenes. In *International Conference on Learning Representations*, 2025.

[10] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, et al. Video generation models as world simulators. *OpenAI Blog*, 2024.

[11] Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural SLAM. *arXiv preprint arXiv:2004.05155*, 2020.

[12] Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *Advances in Neural Information Processing Systems*, 37:24081–24125, 2024.

[13] Tao Chen, Saurabh Gupta, and Abhinav Gupta. Learning exploration policies for navigation. *arXiv preprint arXiv:1903.01959*, 2019.

[14] Ethan Chern, Jiadi Su, Yan Ma, and Pengfei Liu. Anole: An open, autoregressive, native large multimodal models for interleaved image-text generation. *arXiv preprint arXiv:2407.06135*, 2024.

[15] Haozhuang Chi, Jingsong Liang, Ziying Song, Lei Yang, Shihao Li, Haoruo Zhang, and Chen Lv. Pv-wm: A heterogeneous micro-macro world model for articulated pedestrian-vehicle co-rollout. *arXiv preprint arXiv:2609.07328*, 2026.

[16] Haozhuang Chi, Daosheng Qiu, Hao Su, Haochen Liu, Zirui Li, Haoruo Zhang, and Chen Lv. Driver-wm: A driver-centric traffic-conditioned latent world model for in-cabin dynamics rollout. In *European Conference on Computer Vision*, pages 402–421. Springer, 2026.

[17] Meng Chu, Xuan Billy Zhang, Kevin Qinghong Lin, Lingdong Kong, Jize Zhang, Teng Tu, Weijian Ma, Ziqi Huang, Senqiao Yang, Wei Huang, et al. Agentic world modeling: Foundations, capabilities, laws, and beyond. *arXiv preprint arXiv:2604.22748*, 2026.

[18] Branton DeMoss, Paul Duckworth, Nick Hawes, and Ingmar Posner. DITTO: Offline imitation learning with world models. *arXiv preprint arXiv:2302.03086*, 2023.

[19] Yijie Deng, Shuaihang Yuan, Geeta Chandra Raju Bethala, and Anthony Tzes. Hierarchical scoring with 3d gaussian splatting for instance image-goal navigation. *IEEE Robotics and Automation Letters*, 2026.

[20] Yijie Deng, Shuaihang Yuan, and Yi Fang. Anyimagenav: Any-view geometry for precise last-meter image-goal navigation. *arXiv preprint arXiv:2604.05351*, 2026.

[21] Yijie Deng, Shuaihang Yuan, Congcong Wen, Hao Huang, Anthony Tzes, Geeta Chandra Raju Bethala, and Yi Fang. Mapbert: Bitwise masked modeling for real-time semantic mapping generation. *arXiv preprint arXiv:2506.07350*, 2025.

[22] Jingtao Ding, Yunke Zhang, Yu Shang, Yuheng Zhang, Zefang Zong, Jie Feng, Yuan Yuan, Hongyuan Su, Nian Li, Nicholas Sukiennik, et al. Understanding world or predicting future? a comprehensive survey of world models. *ACM Computing Surveys*, 2024.

[23] Yifei Dong, Fengyi Wu, Guangyu Chen, Lingdong Kong, Qiyu Hu, Yuxuan Zhou, Xu Zhu, Jingdong Sun, Jun-Yan He, Qi Dai, Alexander G. Hauptmann, and Zhi-Qi Cheng. Towards unified world models for visual navigation via memory-augmented planning and foresight. In *European Conference on Computer Vision*, pages 276–295. Springer, 2026.

[24] Yifei Dong, Fengyi Wu, Qi He, Lingdong Kong, Heng Li, Minghan Li, Zebang Cheng, Yuxuan Zhou, Jingdong Sun, Qi Dai, Alexander G. Hauptmann, and Zhi-Qi Cheng. HA-VLN 2.0: An open benchmark and leaderboard for human-aware navigation in discrete and continuous environments with dynamic multi-human interactions. *arXiv preprint arXiv:2503.14229*, 2025.

[25] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[26] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. GPTQ: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022.

[27] J Frey, M Mattamala, N Chebrolu, C Cadena, M Fallon, and M Hutter. Fast traversability estimation for wild visual navigation. *Robotics: Science and Systems Proceedings*, 19, 2023.

[28] Stephanie Fu, Netanel Tamir, Shobhita Sundaram, Lucy Chai, Richard Zhang, Tali Dekel, and Phillip Isola. DreamSim: Learning new dimensions of human visual similarity using synthetic data. *arXiv preprint arXiv:2306.09344*, 2023.

[29] Zipeng Fu, Ashish Kumar, Ananye Agarwal, Haozhi Qi, Jitendra Malik, and Deepak Pathak. Coupling vision and proprioception for navigation of legged robots. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17273–17283, 2022.

[30] Oran Gafni, Adam Polyak, Oron Ashual, Shelly Sheynin, Devi Parikh, and Yaniv Taigman. Make-a-scene: Scene-based text-to-image generation with human priors. In *European Conference on Computer Vision*, pages 89–106. Springer, 2022.

[31] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4D: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18995–19012, 2022.

[32] David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

[33] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pages 1861–1870. PMLR, 2018.

[34] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

[35] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering Atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

[36] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

[37] Noriaki Hirose, Amir Sadeghian, Marynel Vázquez, Patrick Goebel, and Silvio Savarese. GONet: A semi-supervised deep learning approach for traversability estimation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3044–3051, 2018.

[38] Noriaki Hirose, Dhruv Shah, Ajay Sridhar, and Sergey Levine. SACSoN: Scalable autonomous control for social navigation. *IEEE Robotics and Automation Letters*, 9(1):49–56, 2023.

[39] Alain Hore and Djemel Ziou. Image quality metrics: PSNR vs. SSIM. In *IEEE International Conference on Pattern Recognition*, pages 2366–2369, 2010.

[40] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. LoRA: Low-rank adaptation of large language models. *International Conference on Learning Representations*, 2022.

[41] Haresh Karnan, Anirudh Nair, Xuesu Xiao, Garrett Warnell, Sören Pirk, Alexander Toshev, Justin Hart, Joydeep Biswas, and Peter Stone. Socially compliant navigation dataset (SCAND): A large-scale dataset of demonstrations for social navigation. *IEEE Robotics and Automation Letters*, 7(4):11807–11814, 2022.

[42] Efstathios Karypidis, Ioannis Kakogeorgiou, Spyros Gidaris, and Nikos Komodakis. DINO-Foresight: Looking into the future with DINO. *arXiv preprint arXiv:2412.11673*, 2024.

[43] Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[44] Jing Yu Koh, Honglak Lee, Yinfei Yang, Jason Baldridge, and Peter Anderson. PathDreamer: A world model for indoor navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14738–14748, 2021.

[45] Zahra Kolagar and Alessandra Zarcone. Aligning uncertainty: Leveraging llms to analyze uncertainty transfer in text summarization. In *Workshop on Uncertainty-Aware NLP*, pages 41–61, 2024.

[46] Lingdong Kong, Shaoyuan Xie, Zeying Gong, Ye Li, Meng Chu, Ao Liang, Yuhao Dong, Tianshuai Hu, Ronghe Qiu, Rong Li, Hanjiang Hu, Dongyue Lu, Wei Yin, Wenhao Ding, Linfeng Li, Hang Song, Wenwei Zhang, Yuexin Ma, et al. The RoboSense challenge: Sense anything, navigate anywhere, adapt across platforms. *arXiv preprint arXiv:2601.05014*, 2026.

[47] Lingdong Kong, Wesley Yang, Jianbiao Mei, Youquan Liu, Ao Liang, Dekai Zhu, Dongyue Lu, Wei Yin, Xiaotao Hu, Mingkai Jia, Junyuan Deng, Kaiwen Zhang, Yang Wu, Tianyi Yan, Shenyuan Gao, Song Wang, Linfeng Li, Liang Pan, Yong Liu, Jianke Zhu, Wei Tsang Ooi, Steven C. H. Hoi, and Ziwei Liu. 3D and 4D world modeling: A survey. *arXiv preprint arXiv:2509.07996*, 2025.

[48] Xianghao Kong, Jinyu Chen, Wenguan Wang, Hang Su, Xiaolin Hu, Yi Yang, and Si Liu. Controllable navigation instruction generation with chain of thought prompting. In *European Conference on Computer Vision*, pages 37–54. Springer, 2024.

[49] Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the Nav-Graph: Vision-and-language navigation in continuous environments. In *European Conference on Computer Vision*, pages 104–120. Springer, 2020.

[50] Boyi Li, Yifan Shen, Yuanzhe Liu, Yifan Xu, Jiateng Liu, Xinzhuo Li, Zhengyuan Li, Jingyuan Zhu, Yunhan Zhong, Fangzhou Lan, et al. Toward cognitive supersensing in multimodal large language model. *arXiv preprint arXiv:2602.01541*, 2026.

[51] Chengzu Li, Wenshan Wu, Huanyu Zhang, Yan Xia, Shaoguang Mao, Li Dong, Ivan Vulić, and Furu Wei. Imagine while reasoning in space: Multimodal visualization-of-thought. *arXiv preprint arXiv:2501.07542*, 2025.

[52] Heng Li, Minghan Li, Zhi-Qi Cheng, Yifei Dong, Yuxuan Zhou, Jun-Yan He, Qi Dai, Teruko Mitamura, and Alexander G Hauptmann. Human-aware vision-and-language navigation: Bridging simulation to reality with dynamic human interactions. *Advances in Neural Information Processing Systems*, 37:119411–119442, 2024.

[53] Ao Liang, Lingdong Kong, Tianyi Yan, Hongsi Liu, Wesley Yang, Ziqi Huang, Wei Yin, Jialong Zuo, Yixuan Hu, Dekai Zhu, Dongyue Lu, Youquan Liu, Guangfeng Jiang, Linfeng Li, Xiangtai Li, Long Zhuo, Lai Xing Ng, Benoit R. Cottereau, Changxin Gao, Liang Pan, Wei Tsang Ooi, and Ziwei Liu. WorldLens: Full-spectrum evaluations of driving world models in real world. *arXiv preprint arXiv:2512.10958*, 2025.

[54] Ao Liang, Youquan Liu, Yu Yang, Dongyue Lu, Linfeng Li, Lingdong Kong, Huaici Zhao, and Wei Tsang Ooi. LiDARCrafter: Dynamic 4D world modeling from LiDAR sequences. In *AAAI Conference on Artificial Intelligence*, volume 40, 2026.

[55] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[56] Bingqian Lin, Yunshuang Nie, Ziming Wei, Jiaqi Chen, Shikui Ma, Jianhua Han, Hang Xu, Xiaojun Chang, and Xiaodan Liang. NavCoT: Boosting LLM-based vision-and-language navigation via learning disentangled reasoning. *arXiv preprint arXiv:2403.07376*, 2024.

[57] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in Neural Information Processing Systems*, 36:34892–34916, 2023.

[58] Mengmeng Liu, Jiuming Liu, Michael Ying Yang, Chaokang Jiang, Jiangtao Li, Yunpeng Zhang, Hesheng Wang, Francesco Nex, and Hao Cheng. Streamvlo: Streaming visual-lidar odometry with cumulative drift compensation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 39086–39097, 2026.

[59] Mengmeng Liu, Diankun Zhang, Jiuming Liu, Jianfeng Cui, Hongwei Xie, Guang Chen, Hangjun Ye, Francesco Nex, Hao Cheng, and Michael Ying Yang. Universe: Unified video action models for autonomous driving with flexible mask-modulated modality generation. *arXiv preprint arXiv:2607.05133*, 2026.

[60] Mengmeng Liu, Diankun Zhang, Jiuming Liu, Jianfeng Cui, Hongwei Xie, Guang Chen, Hangjun Ye, Michael Ying Yang, Francesco Nex, and Hao Cheng. Driveva: Video action models are zero-shot drivers. In *European Conference on Computer Vision*, pages 315–335. Springer, 2026.

[61] Yuanzhe Liu, Jingyuan Zhu, Yuchen Mo, Gen Li, Xu Cao, Jin Jin, Yifan Shen, Zhengyuan Li, Tianjiao Yu, Wenzhen Yuan, et al. Palm: Progress-aware policy learning via affordance reasoning for long-horizon robotic manipulation. *arXiv preprint arXiv:2601.07060*, 2026.

[62] Yuze Liu, Tingjie Liu, Tiehua Zhang, Youhua Xia, Jinze Wang, Zhishu Shen, Jiong Jin, Zhijun Ding, and Fei Richard Yu. Grl-prompt: Towards prompts optimization via graph-empowered reinforcement learning using llms' feedback. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 426–438. Springer, 2025.

[63] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[64] Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. Learning latent plans from play. In *Conference on Robot Learning*, pages 1113–1132. PMLR, 2020.

[65] Oier Mees, Lukas Hermann, and Wolfram Burgard. What matters in language conditioned robotic imitation learning over unstructured data. *IEEE Robotics and Automation Letters*, 7(4):11205–11212, 2022.

[66] Piotr Mirowski, Razvan Pascanu, Fabio Viola, Hubert Soyer, Andrew J Ballard, Andrea Banino, Misha Denil, Ross Goroshin, Laurent Sifre, Koray Kavukcuoglu, et al. Learning to navigate in complex environments. *arXiv preprint arXiv:1611.03673*, 2016.

[67] Iman Nematollahi, Branton DeMoss, Akshay L Chandra, Nick Hawes, Wolfram Burgard, and Ingmar Posner. LUMOS: Language-conditioned imitation learning with world models. In *IEEE International Conference on Robotics and Automation*, 2025.

[68] Deepak Pathak, Parsa Mahmoudieh, Guanghao Luo, Pulkit Agrawal, Dian Chen, Yide Shentu, Evan Shelhamer, Jitendra Malik, Alexei A Efros, and Trevor Darrell. Zero-shot visual imitation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 2050–2053, 2018.

[69] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023.

[70] Yuankai Qi, Qi Wu, Peter Anderson, Xin Wang, William Yang Wang, Chunhua Shen, and Anton van den Hengel. Reverie: Remote embodied visual referring expression in real indoor environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9982–9991, 2020.

[71] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[72] Dhruv Shah, Benjamin Eysenbach, Gregory Kahn, Nicholas Rhinehart, and Sergey Levine. Rapid exploration for open-world navigation with latent goal models. *arXiv preprint arXiv:2104.05859*, 2021.

[73] Dhruv Shah, Ajay Sridhar, Arjun Bhorkar, Noriaki Hirose, and Sergey Levine. GNM: A general navigation model to drive any robot. *arXiv preprint arXiv:2210.03370*, 2022.

[74] Dhruv Shah, Ajay Sridhar, Nitish Dashora, Kyle Stachowicz, Kevin Black, Noriaki Hirose, and Sergey Levine. ViNT: A foundation model for visual navigation. *arXiv preprint arXiv:2306.14846*, 2023.

[75] Yifan Shen, Yuanzhe Liu, Jingyuan Zhu, Xu Cao, Xiaofeng Zhang, Yixiao He, Wenming Ye, James Rehg, and Ismini Lourentzou. Fine-grained preference optimization improves spatial reasoning in vlms. *Advances in Neural Information Processing Systems*, 38:17929–17960, 2026.

[76] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[77] Kiwhan Song, Boyuan Chen, Max Simchowitz, Yilun Du, Russ Tedrake, and Vincent Sitzmann. History-guided video diffusion. *arXiv preprint arXiv:2502.06764*, 2025.

[78] Ajay Sridhar, Dhruv Shah, Catherine Glossop, and Sergey Levine. NoMaD: Goal masked diffusion policies for navigation and exploration. In *IEEE International Conference on Robotics and Automation*, pages 63–70, 2024.

[79] Jürgen Sturm, Wolfram Burgard, and Daniel Cremers. Evaluating egomotion and structure-from-motion approaches using the TUM RGB-D benchmark. In *IEEE/RSJ International Conference on Intelligent Robot Systems Workshops*, volume 13, page 6, 2012.

[80] Xiaole Tang, Xiaoyi He, Jiayi Xu, Xiang Gu, and Jian Sun. Learning continuous wasserstein barycenter space for generalized all-in-one image restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026.

[81] Xiaole Tang, Jiayi Xu, Xiang Gu, Yan Yang, and Jian Sun. Binding multiple modalities via multimodal wasserstein barycenter. *arXiv preprint arXiv:2609.33800*, 2026.

[82] Chameleon Team. Chameleon: Mixed-modal early-fusion foundation models. *arXiv preprint arXiv:2405.09818*, 2024.

[83] Qwen Team. Qwen3.5: Accelerating productivity with native multimodal agents, February 2026.

[84] Shengbang Tong, David Fan, Jiachen Li, Yunyang Xiong, Xinlei Chen, Koustuv Sinha, Michael Rabbat, Yann LeCun, Saining Xie, and Zhuang Liu. MetaMorph: Multimodal understanding and generation via instruction tuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17001–17012, 2025.

[85] 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 *IEEE International Conference on Robotics and Automation*, pages 2546–2552, 2022.

[86] Dani Valevski, Yaniv Leviathan, Moab Arar, and Shlomi Fruchter. Diffusion models are real-time game engines. *arXiv preprint arXiv:2408.14837*, 2024.

[87] Fu-Yun Wang, Zhaoyang Huang, Alexander Bergman, Dazhong Shen, Peng Gao, Michael Lingelbach, Keqiang Sun, Weikang Bian, Guanglu Song, Yu Liu, et al. Phased consistency models. *Advances in Neural Information Processing Systems*, 37:83951–84009, 2024.

[88] Jinze Wang, Lu Zhang, Yiyang Cui, Tiehua Zhang, Zhishu Shen, Yuze Liu, Xingjun Ma, and Jiong Jin. Do we really need sft? prompt-as-policy over knowledge graphs for cold-start next poi recommendation. *arXiv preprint arXiv:2510.08012*, 2025.

[89] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[90] Fengyi Wu, Yifei Dong, Zhi-Qi Cheng, Yilong Dai, Guangyu Chen, Hang Wang, Qi Dai, and Alexander G Hauptmann. GoViG: Goal-conditioned visual navigation instruction generation. *arXiv preprint arXiv:2508.09547*, 2025.

[91] Amber Xie, Oleh Rybkin, Dorsa Sadigh, and Chelsea Finn. Latent diffusion planning for imitation learning. *arXiv preprint arXiv:2504.16925*, 2025.

[92] Eric Xing, Mingkai Deng, Jinyu Hou, and Zhiting Hu. Critiques of world models. *arXiv preprint arXiv:2507.05169*, 2025.

[93] Karmesh Yadav, Jacob Krantz, Ram Ramrakhya, Santhosh Kumar Ramakrishnan, Jimmy Yang, Austin Wang, John Turner, Aaron Gokaslan, Vincent-Pierre Berges, Roozbeh Mootaghi, Oleksandr Maksymets, Angel X Chang, Manolis Savva, Alexander Clegg, Devendra Singh Chaplot, and Dhruv Batra. Habitat challenge 2023. https://aihabitat.org/challenge/2023, 2023.

[94] Xuan Yao, Junyu Gao, and Changsheng Xu. NavMorph: A self-evolving world model for vision-and-language navigation in continuous environments. *arXiv preprint arXiv:2506.23468*, 2025.

[95] Jiwen Yu, Yiran Qin, Xintao Wang, Pengfei Wan, Di Zhang, and Xihui Liu. GameFactory: Creating new games with generative interactive videos. *arXiv preprint arXiv:2501.08325*, 2025.

[96] Tianjiao Yu, Xinzhuo Li, Yifan Shen, Onkar Susladkar, Yuanzhe Liu, Xiaona Zhou, and Ismini Lourentzou. Elsa3d: Elastic semantic anchoring for unified 3d understanding and generation. *arXiv preprint arXiv:2607.06565*, 2026.

[97] Edwin Zhang, Yujie Lu, William Wang, and Amy Zhang. Language control diffusion: Efficiently scaling through space, time, and tasks. *arXiv preprint arXiv:2210.15629*, 2022.

[98] 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/CVF Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018.

[99] Guosheng Zhao, Xiaofeng Wang, Zheng Zhu, Xinze Chen, Guan Huang, Xiaoyi Bao, and Xingang Wang. DriveDreamer-2: LLM-enhanced world models for diverse driving video generation. In *AAAI Conference on Artificial Intelligence*, volume 39, pages 10412–10420, 2025.

[100] Zheng Zhu, Xiaofeng Wang, Wangbo Zhao, Chen Min, Nianchen Deng, Min Dou, Yuqi Wang, Botian Shi, Kai Wang, Chi Zhang, et al. Is Sora a world simulator? a comprehensive survey on general world models and beyond. *arXiv preprint arXiv:2405.03520*, 2024.

Appendix for Language-Conditioned World Modeling for Visual Navigation

This supplementary material provides additional details and results that complement the main paper. Sec. A presents multi-style data samples from the LCVN dataset. Sec. B includes pseudo-code for the LCVN-WM and LCVN-AC, as well as prompt and training details for the LCVN-Uni. Sec. C reports further experimental details and more experiments.

A LCVN Dataset Details

As shown in Fig. A1, this section complements Sec. 3.1 with details on the trajectory sources, the seeded few-shot annotation pipeline, human evaluation, and qualitative samples of the three instruction styles.

A.1 Trajectory Sources and Preprocessing

Go Stanford, ReCon, SCAND, and HuRoN provide diverse indoor and structured outdoor trajectories that populate the training and val seen splits, while TartanDrive contributes exclusively to val unseen and partially to test. TartanDrive's off-road driving scenes differ markedly from the other four sources in terrain, camera dynamics, and visual appearance, providing a strong distribution shift for zero-shot evaluation. The test split intentionally blends held-out trajectories from all five sources so that a single evaluation simultaneously measures in-domain proficiency and out-of-domain generalization. To standardize trajectories from heterogeneous sources, we normalize per-frame displacement by each source's average step size to yield an embodiment-agnostic continuous control space, filter out backward movements and trajectories shorter than three steps following [7, 78], and segment each visual stream into semantically coherent sub-scenes using Qwen-3.5-9B [83].

A.2 Seeded Few-Shot Annotation Pipeline

Our annotation team first manually writes a seed set covering the three target styles, distributed across diverse environment types and trajectory complexities (straight, single-turn, multi-turn). Annotators are instructed to write naturally and idiomatically, with attention to stylistic markers that distinguish the three styles: terse imperatives for Concise, descriptive subordinate clauses for Intricate, and landmark-anchored prepositional phrases for Landmark-based. This seed set serves as the linguistic anchor for all subsequent generation.

For each target trajectory, we render a top-down 2D map alongside the egocentric video. The map plots the agent's planar trajectory using local odometry, with the start point marked, the heading direction indicated by an arrow at each waypoint, and turning angles annotated; sampled camera frustums convey the egocentric viewpoint orientation. Without this 2D map as visual input, the VLM frequently misinterprets terrain-induced camera shake or local sidesteps as deliberate global heading changes, producing instructions like "turn right" for trajectories that are globally straight. The 2D map disambiguates global heading from local motion noise, substantially improving geometric correctness of generated instructions.

We then construct a style-specific few-shot prompt for each trajectory by sampling seeds matched to the target style, environment type, and complexity, and pass this prompt along with the target trajectory's egocentric frames and 2D map to Qwen-3.5-9B. We generate three candidate instructions per trajectory (one per style). All generated instructions are subsequently reviewed by our annotation team, who cross-reference them with the corresponding video and 2D map and check for geometric correctness, landmark accuracy, style adherence, and naturalness. Annotators correct geometric or landmark errors and rewrite instructions that drift from the target style. The final dataset thus combines human authorship at the seed stage, VLM scaling at the generation stage, and human verification at the refinement stage, rather than the simple "machine-generated then verified" pipeline that motivates concerns about linguistic homogeneity.

A.3 Human Quality Evaluation

To quantitatively assess the quality of the final instructions, we conduct a human evaluation on a random subset of the dataset. We sample 640 trajectories roughly balanced across the five sources and

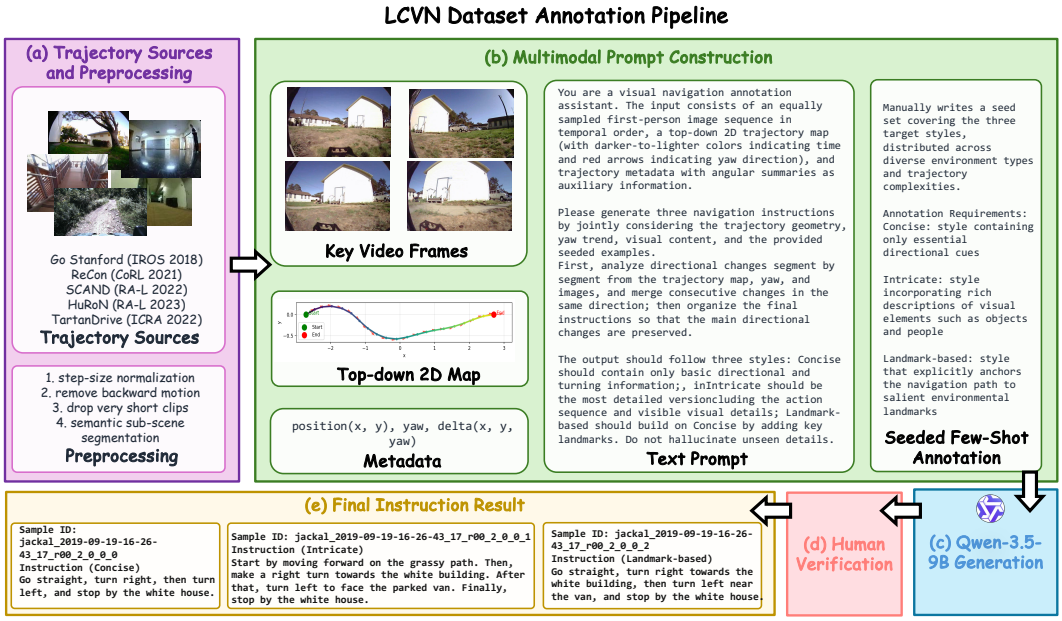

Figure A1: Overview of **annotation pipeline** of the LCVN dataset.

three instruction styles, and recruit volunteer evaluators from our research group who are not involved in the annotation pipeline. Each evaluator watches the egocentric video, views the corresponding 2D trajectory map, and rates the paired instruction on five dimensions using a 1–6 Likert scale, where 6 indicates the instruction is indistinguishable from expert human-written ones: (1) *Trajectory–Instruction Alignment*, whether the instruction accurately describes the path including turns, distances, and stopping point; (2) *Style Adherence*, whether the instruction conforms to its target style without drifting into characteristics of another; (3) *Geometric Correctness*, whether directional references and spatial relations are consistent with egocentric viewpoint; (4) *Landmark Accuracy*, whether mentioned visual landmarks are present and correctly localized; and (5) *Naturalness & Fluency*, whether the instruction reads as natural language a human guide would produce. Each trajectory–instruction pair is rated by three evaluators, and scores are averaged across evaluators and dimensions.

The overall mean score across all 640 samples is 5.2 out of 6.0, indicating that the vast majority of instructions are rated close to expert human quality. These results confirm that the combination of seeded few-shot generation, 2D map conditioning, and human verification yields instructions of consistently high quality across all three styles.

A.4 Multi-Style Instruction Examples

Table A1 presents two representative trajectories, each annotated in all three styles, with text in purple marking directional guidance and text in blue marking landmark references. The Concise instructions provide minimal yet sufficient directional guidance, the Intricate instructions enrich the path with descriptions of surrounding objects and scene context, and the Landmark-based instructions explicitly tie actions to recognizable environmental features. The three styles are designed to be complementary rather than redundant: Concise probes whether models can follow minimal directional language, Intricate probes whether models can extract navigationally-relevant content from verbose descriptions, and Landmark-based probes whether models can ground language into specific visual features. Comparing model performance across styles in Table 4 thus serves as a structured diagnostic of where language grounding succeeds or fails.

Generated Annotations and Human-Written Seeds. To illustrate how the seeded annotation pipeline preserves these stylistic conventions, we compare a generated annotation with a human-written seed from the same source dataset. The examples describe different trajectories.

Generated annotation.

Table A1: **Multi-style Instruction Samples from the LCVN Dataset.** Text in purple denotes directional guidance, while text in blue marks landmark references. Each trajectory in LCVN is paired with three distinct instruction styles (concise, intricate and landmark-grounded), enabling a comprehensive evaluation of model generalization across varying levels of specificity and emphasis [45].

Trajectory example 1 (concise style). Walk straight ahead and stop at the intersection.
Trajectory example 1 (intricate style). You are on a wide sidewalk lined with trees on both sides. A pedestrian is approaching from the front. Continue walking until a red stall appears on your right, with three people gathered around it, and stop in front of the stall.
Trajectory example 1 (landmark-grounded style). Walk along the sidewalk until you reach a red stall ahead, and stop in front of it.
Trajectory example 2 (concise style). Walk straight ahead, turn left at the first corner, and stop along the wall.
Trajectory example 2 (intricate style). Proceed straight along the gray corridor, passing a row of white cabinets and a glass-walled office on your left where people appear to be working. Continue forward until you reach a gray door, with a small white trash bin positioned to its left. Then turn left into another gray corridor and stop along the wall on your left.
Trajectory example 2 (landmark-grounded style). Walk straight down the corridor, passing a glass-walled office on your left. Continue until you reach a gray door, then turn left and stop along the wall on your left.

Concise: "Move forward, turn right, and continue straight until you reach the corner near the window and desk, then stop."

Intricate: "Start by moving forward past a wooden nightstand and a bed with white linens. As you proceed, turn right to face a large glass sliding door leading to a balcony. Continue walking straight across the light wooden floor, passing the foot of the bed on your left and approaching a white desk with a black top on your right. Stop near the corner where the desk meets the wall by the window."

Landmark-based: "Walk past the bed and nightstand, then turn right towards the glass sliding doors. Continue straight across the room, passing the foot of the bed, and stop near the white desk located by the window."

Human-written seed.

Concise: "Turn right into the bathroom, walk straight ahead then stop."

Intricate: "In an empty bedroom, on the left is a bright window and many household items. On the right is a bathroom with a yellow light. Turn right and enter the bathroom. After entering the bathroom, there is a toilet on the left, and there is a mirror and a dressing table ahead. Finally, stop in front of a trash can."

Landmark-based: "Starting from an open space, turn right to go into the bathroom. Walk towards the trash can and stop in front of it."

Both examples exhibit the intended style-specific patterns: short action sequences in Concise, detailed scene descriptions in Intricate, and landmark-anchored directions in Landmark-based. This qualitative correspondence is consistent with the seeded few-shot design.

B Additional Implementation Details

B.1 Pseudo-Code for LCVN-WM & LCVN-AC

Fig. 2 in the main text illustrates the overall architecture of both LCVN-WM and LCVN-AC, highlighting three stages of training and inference. To complement this high-level overview, we provide detailed pseudo-code here (Algorithms 1, 2 and 3) that specifies the step-by-step procedures underlying each stage.

Algorithm 1: Training Phase 1: LCVN-WM with diffusion forcing

Input: Trajectory observations $o_{0:n}$, actions $a_{0:n-1}$, instruction I, timeshift t_s, VAE encoder Enc, language encoder CLIPTextEnc, MLP layer MLP, world model F_θ, context size k

while *not converged* **do**

  Sample a trajectory $(o_{0:n}, a_{0:n-1}, I, t_s)$

  for $i \leftarrow 0$ **to** n **do**

    $s_i \leftarrow \text{Enc}(o_i)$ `// encode images to latents`

  $I_{\text{clip}} \leftarrow \text{CLIPTextEnc}(I)$ $\mathcal{L}_{\text{WM}} \leftarrow 0$

  for $t \leftarrow k-1$ **to** $n-1$ **do**

    Build context window $\mathbf{s}_t \leftarrow [s_t, s_{t-1}, \ldots, s_{t-k+1}]$

    `// Diffusion Forcing`

    for $m \leftarrow 0$ **to** $k-1$ **do**

      Sample noise level ℓ_m and $\epsilon_m \sim \mathcal{N}(0, I)$

      $\tilde{s}_{t-m} \leftarrow \sqrt{\alpha_{\ell_m}}\, s_{t-m} + \sqrt{1-\alpha_{\ell_m}}\, \epsilon_m$ (Eq. 2)

    $\tilde{\mathbf{s}}_t \leftarrow [\tilde{s}_t, \tilde{s}_{t-1}, \ldots, \tilde{s}_{t-k+1}]$

    $e_a \leftarrow \text{MLP}(a_t)$; $e_{ts} \leftarrow \text{MLP}(t_s)$

    $e_d \leftarrow \text{MLP}(t_{\text{diff}})$

    $e_{\text{cond}} \leftarrow e_a + e_{ts} + e_d$

    $\hat{s}_{t+1} \leftarrow F_\theta(\tilde{\mathbf{s}}_t, I_{\text{clip}}, e_{\text{cond}})$

    $\mathcal{L}_{\text{WM}} \mathrel{+}= \mathcal{L}_{\text{diff}}(\hat{s}_{t+1}, s_{t+1}, t_{\text{diff}})$

  Update θ

Training Phase 1 (LCVN-WM). Algorithm 1 outlines the training procedure of the LCVN-WM. LCVN-WM incorporates language conditioning by extending a diffusion transformer with cross-attention to CLIP-encoded instructions, together with action and time-shift conditioning. In addition, it adopts Diffusion Forcing, which applies heterogeneous noise levels across latent context window. This mechanism strengthens temporal modeling and enables the model to predict future latent states conditioned on actions and instructions.

Training Phase 2 (LCVN-AC). Algorithm 2 details the actor–critic agent trained entirely in the latent space of LCVN-WM. The learner encoder produces latent plans conditioned on current states and instructions, which are aligned with expert plans via KL divergence. Actor–critic optimization incorporates intrinsic rewards that measure agreement between predicted and expert rollouts, while the critic regresses to λ-returns. The actor is regularized by both plan consistency and language alignment losses, ensuring semantically grounded navigation behavior.

Inference Stage. Algorithm 3 outlines the coupled inference procedure. Given an initial observation and instruction, LCVN-WM imagines future latent states, while LCVN-AC generates corresponding actions conditioned on both the imagined states and instruction embeddings. The agent executes actions until a stop command is issued, thereby integrating language grounding, imagination, and policy learning into a unified navigation loop.

Together, these pseudo-code listings complement the schematic in Fig. 2, clarifying how LCVN-WM and LCVN-AC interact across training and inference to realize language-conditioned visual navigation.

B.2 Prompt Design for LCVN-Uni

We examine detailed prompt formulation and response behaviors of coupling navigation planning and visualization in Fig. A2. This example illustrates how multimodal inputs including navigation instructions and egocentric observations jointly guide both planner and world model in LCVN task.

B.3 Training Loss Details for LCVN-Uni

LCVN-Uni optimizes a unified objective that jointly learns navigation planning and world modeling through a combined loss function. At each training step, the model outputs logits for both action bins

Algorithm 2: Training Phase 2: LCVN-AC with Intrinsic Rewards

Input: Frozen LCVN-WM F_θ, latent observations $s^E_{0:n}$, actions $a_{0:n-1}$, instruction I, timeshift t_s, context size k, horizon H
Input: Expert encoder q_{expert}, learner encoder q_{learner}, actor π_θ, critic v_ψ
while *not converged* **do**
  Sample a trajectory $(s^E_{0:n}, a_{0:n-1}, I)$
  $I_{\text{clip}} \leftarrow \text{CLIPTextEnc}(I)$
  Initialize losses: $\mathcal{L}_{\text{critic}} \leftarrow 0,\ \mathcal{L}_{\text{actor}} \leftarrow 0$
  for $t \leftarrow 0$ **to** $n-1$ **do**
    Expert plan: $z_{\text{expert}} \sim q_{\text{expert}}(z \mid s^E_{t:n}, I_{\text{clip}})$
    Learner plan: $z_{\text{plan}} \sim q_{\text{learner}}(z \mid s^E_t, I_{\text{clip}})$
    Compute KL divergence (Eq. 3): $\mathcal{L}^t_{\text{KL}} \leftarrow D_{\text{KL}}(q_{\text{expert}} \,\|\, q_{\text{learner}})$
    Build latent history $\mathbf{s}_t \leftarrow [s^E_t, s^E_{t-1}, \ldots, s^E_{t-k}]$ (pad if $t < k$)
    for $k \leftarrow t$ **to** $\min(t+H-1, n-1)$ **do**
      Sample action: $\hat{a}_k \sim \pi_\theta(s^E_k, I_{\text{clip}}, z_{\text{plan}})$
      Predict next latent with WM (Eq. 4):
        $\hat{s}^\pi_{k+1} \sim F_\theta(\hat{s}^\pi_{k+1} \mid \mathbf{s}_k, \hat{a}_k, I_{\text{clip}}, t_s)$
      Compute intrinsic reward r^{int}_k (Eq. 5)
      Append r^{int}_k, $\hat{s}^\pi_{k+1}$; update context $\mathbf{s}_{k+1}$
    Compute discounted returns (Eq. 6)
    for *each rollout step* k **do**
      Compute critic loss $\mathcal{L}_{\text{critic}}$ (Eq. 7)
      Compute actor loss $\mathcal{L}_{\text{actor}}$ (Eq. 8)
  Update parameters ψ, θ, q_{learner}

and visual tokens, enabling end-to-end optimization:

$$\mathcal{L}_{\text{joint}} = \mathcal{L}_{\text{plan}} + \lambda \mathcal{L}_{\text{imagine}}, \tag{A1}$$

where $\mathcal{L}_{\text{plan}}$ is the discretized bin-token loss [23] for action prediction, and $\mathcal{L}_{\text{imagine}}$ [23, 51] enforces fidelity in predicted future observations. The hyperparameter λ balances the relative importance between planning and imagination.

Following the bin tokenization scheme, each navigation action $a_t \in \mathbb{R}^3$ is represented as (x_t, y_t, ϕ_t) and discretized into tokens drawn from disjoint sets $\mathcal{B}_x$, $\mathcal{B}_y$, and $\mathcal{B}_\phi$. Let $P(s_j)$ denote the predicted probability distribution over vocabulary tokens at position j. The action loss is defined as:

$$\mathcal{L}_{\text{plan}} = \frac{1}{3} \sum\nolimits_{d \in \{x,y,\phi\}} \Big(-\log P(b^*_d \,|\, s_j \in \mathcal{B}_d) \Big) + \mathcal{L}_{\text{text}}, \tag{A2}$$

where b^*_d is the ground-truth bin token for dimension d, and $\mathcal{L}_{\text{text}}$ is the cross-entropy loss for text tokens. For visual prediction, given ground-truth embedding $\mathbf{u}_j$ at token position j (out of m tokens in the predicted observation $\hat{o}_{t+1}$) and a codebook $\mathcal{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$ of size K, imagination loss is:

$$\mathcal{L}_{\text{imagine}} = \frac{1}{m} \sum\nolimits_{j=1}^{m} \sum\nolimits_{k=1}^{K} \|\mathbf{u}_j - \mathbf{c}_k\|^2 \cdot P(s_j = \mathbf{c}_k), \tag{A3}$$

where $\|\mathbf{u}_j - \mathbf{c}_k\|^2$ is the squared distance between $\mathbf{u}_j$ and codebook entry $\mathbf{c}_k$, and $P(s_j = \mathbf{c}_k)$ is the predicted probability of assigning token j to codebook entry k. This joint optimization allows LCVN-Uni to learn shared multimodal representations that simultaneously support accurate navigation planning and visual imagination.

C Additional Experiments & Results

C.1 Evaluation Metric Details

We evaluate overall system performance using two complementary categories of metrics in this work:

Algorithm 3: Inference with coupling LCVN-WM and LCVN-AC

Input: VAE encoder Enc, LCVN-WM F_θ, learner encoder q_{learner}, actor π_ϕ
Input: Instruction I, initial observation o_s, context size k, max steps $T_{\max}$
Encode instruction: $I_{\text{clip}} \leftarrow \text{CLIPTextEnc}(I)$
Encode initial observation: $s_0 \leftarrow \text{Enc}(o_s)$
Initialize context $\mathbf{s}$ (length k) with s_0
Initialize previous action $a_{\text{prev}} \leftarrow \mathbf{0}$
for $t \leftarrow 0$ **to** $T_{\max} - 1$ **do**
  if $t > 0$ **then**
    `// imagine the next visual latent state`
    $\hat{s}_t \sim F_\theta(\hat{s}_t \mid \mathbf{s}, a_{\text{prev}}, I_{\text{clip}}, t_s)$
    Update $\mathbf{s}$ by appending $\hat{s}_t$ and removing the oldest latent
  Sample plan: $z_{\text{plan}} \sim q_{\text{learner}}(z \mid s_t, I_{\text{clip}})$
  `// predict the next action`
  $\hat{a}_t \sim \pi_\phi(\cdot \mid s_t, I_{\text{clip}}, z_{\text{plan}})$
  if $\hat{a}_t = \mathbf{0}$ **then**
    break `// stop action to terminate the process`
  $a_{\text{prev}} \leftarrow \hat{a}_t$
return Executed trajectory $\{\hat{a}_0, \ldots, \hat{a}_T\}$

Navigation Quality. For the LCVN task, we report **Success Rate (SR)** [52, 24], which counts a trajectory as successful if its final distance d to the target is smaller than the agent's average step size $\bar{s}$. Formally, for N trajectories with terminal estimate $\hat{p}_T^{(i)}$ and goal position $p_g^{(i)}$, SR is defined as:

$$\text{SR} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\left[d\left(\hat{p}_T^{(i)},\, p_g^{(i)}\right) < \bar{s} \right]. \tag{A4}$$

We further report **Absolute Trajectory Error (ATE)**, which measures global accuracy by computing the Euclidean distance between aligned predicted and reference poses, and **Relative Pose Error (RPE)**, which captures local consistency by quantifying deviations in relative motion between consecutive estimated and ground-truth poses [79, 58].

Imagination Quality. To evaluate predicted visual rollouts, we employ both structural and perceptual measures: **SSIM** [89], **PSNR** [39], **LPIPS** [98], and **DreamSim** [28]. The latter two are deep perceptual metrics designed to better approximate human judgments. To capture long-horizon stability, we extend these metrics to rollout evaluation, reporting **SSIM**@n, **PSNR**@n, **LPIPS**@n, and **DreamSim**@n. One-step metrics compare the ground-truth next frame o_{t+1} with the one-step prediction $\hat{o}_{t+1}^{(1)}$ obtained from (o_t, a_{t+1}). For horizon n, we perform open-loop rollouts by recursively feeding predicted frames back into the model while conditioning on the ground-truth action sequence $a_{t+1:t+n+1}$. For example, SSIM@n is calculated as:

$$\text{SSIM@}n = \text{SSIM}(o_{t+n},\, \hat{o}_{t+n}^{(n)}), \tag{A5}$$

with analogous definitions for PSNR@n, LPIPS@n, and DreamSim@n. We set n as 8 for experiments.

C.2 Implementation Details

Diamond [2] We select Diamond as one of the world-model baselines for comparison, as it represents a typical UNet-based diffusion world model. We adopt DIAMOND in its best configuration following the official implementation [2]. The diffusion model is trained to autoregressively predict future states at a resolution of 56×56, with an additional upsampler to generate outputs at 224×224. Continuous actions are incorporated through a linear embedding layer for conditioning. We pair Diamond with LCVN-AC to jointly support action prediction and imagination, training LCVN-AC from scratch on LCVN dataset train split, with Diamond replacing LCVN-WM as world model.

NWM [7] We select NWM as another baseline world model for comparison, as it represents a typical DiT-based diffusion architecture. For training, we adopt NWM's best reported configuration: context

LCVN-Uni Prompt

Input

Task: Joint Navigation Single Step Planning and Visualization
Description: Based on a given navigation instruction, the current first-person observation, and starting point observation, predict the next action to take. Also, predict the next first-person view observation after the agent executes this navigation action.
Inputs: Current Action: Move by dx: -0.16, dy: 0.2, dyaw: -0.02
Navigation Instruction: Move across the grass toward the white house on the front-left, and stop in front of it.

Start Observation:



Current Observation:



Action Format: The action can be the language command 'Stop', indicating the end of the trajectory. Alternatively, the action can be shifts composed of three components: - dx: displacement along the agent's facing direction), - dy: displacement perpendicular to the facing direction), - dyaw: change in heading angle (i.e., how much the agent rotates). All components are discretized into bin tokens: for example, - 'dx pos bin 02': dx = +0.02 meters, - 'dy neg bin 23': dy = -0.23 meters, - 'dyaw pos bin 26': counterclockwise rotation of +0.26 radians.
Spatial Interpretation: The magnitude of [dx, dy] reflects how far the agent moves in this step — larger values indicate greater positional shift, leading to larger visual changes. - dyaw controls the agent's rotation. A positive dyaw indicates a left turn (counter-clockwise), while a negative dyaw indicates a right turn (clockwise).
Goal: Predict the most likely next action and next first-person observation, considering how the movement and rotation implied by 'dx', 'dy', and 'dyaw' would affect what the agent sees next.

Response

Predicted action: Move by dx: <dx_neg_bin_26>, dy: <dy_pos_bin_14>, dyaw: <dyaw_pos_bin_05>

Predicted observation:



Figure A2: Prompt design example of LCVN-Uni (context size = 1).

length = 4, goal state = 4, and a CDiT-XL backbone with 1B parameters, training NWM from scratch on the LCVN dataset train split. For inference, we follow the settings in their paper and use 250 diffusion denoising steps to achieve optimal performance. To jointly support action prediction and imagination, we pair NWM with LCVN-AC, training LCVN-AC from scratch on LCVN dataset train split, with NWM replacing LCVN-WM as world model.

LCVN-WM For LCVN-WM, we first train an ImageVAE [43] from scratch to compress input images into 32×32 latent representations. Then we build the model upon a DiT-XL backbone with approximately 1B parameters and a hidden size of 1152. Training leverages diffusion forcing [12], ensuring that each frame retains independent noise levels. We employ the AdamW optimizer [63] with linear warmup and a constant learning rate of 1e-4. For sampling and inference, we adopt the deterministic DDIM sampler [76] with 50 steps following [77].

LCVN-Uni LCVN-Uni is fine-tuned on GAIR Anole-7B [14] with a 4096-token context, while keeping the bin, text and image tokenizers frozen. Input images are resized to 448×448 (height $\times$ width) and discretized into 784 visual tokens when the context size is 1 or 2, and into 625 tokens when

the context size is 4. During training, only the LoRA [40] adapters (rank = 16) in the Transformer's *qkv* projections are updated [57]. Optimization is performed with AdamW for 20 epochs using a learning rate of 2×10^{-4}. Training is conducted on 4×NVIDIA A100 GPUs (80GB each) with a global batch size of 8 (per-GPU batch size = 1, gradient accumulation = 2).

C.3 More Experiments

Inference Speed. Table A2 reports the average inference time per step, including both action prediction and imagination, evaluated on the LCVN val seen split. Inference efficiency is a critical factor for real-world deployment, as embodied agents must operate under strict latency constraints [19]. Compared with NWM [7] and LCVN-Uni, LCVN-WM achieves substantially lower inference time while maintaining competitive performance. Further acceleration is possible through reducing the diffusion denoising steps from 50 to 6 through model distillation [87]. Quantization to 4-bit, which we have not yet explored, is expected to provide additional speedup without significant performance degradation [26]. These results highlight LCVN-WM as a practical and efficient alternative for navigation tasks requiring both accuracy and real-time responsiveness.

Table A2: **Inference time (seconds)** per step, averaged on LCVN val seen split, including both action prediction and imagination. For NWM and LCVN-WM, the times include the latency of LCVN-AC's action prediction.

NWM	LCVN-Uni	LCVN-WM	LCVN-WM (+Distillation.)	LCVN-WM (+Quant. 4-bit)
11.2	20.5	6.4	0.6	0.1 (est. [26])

Interleave *vs.* Predict Both. Table A3 compares two step strategies for LCVN-Uni: Interleave, where action and observation are predicted in alternating substeps, and Predict Both, where both are jointly predicted in a single forward pass. Predict Both achieves comparable navigation and imagination accuracy while offering markedly higher computational efficiency (1.3×), which empirically supports our design choice to adopt joint prediction of actions and observations in the LCVN-Uni agent.

Table A3: **Interleave vs. Predict Both** on navigation and imagination performance of LCVN-Uni (k=2) on LCVN val seen.

	Navigation				Imagination			
Method	**Speed**	**ATE↓**	**RPE↓**	**SR↑**	**SSIM↑**	**DreamSIM↓**	**SSIM@8↑**	**DreamSIM@8↓**
LCVN-Uni (Inter.)	1×	**0.34**	0.12	**0.44**	**0.438**	0.074	0.201	**0.115**
LCVN-Uni	**1.3**×	0.36	**0.11**	0.42	0.423	**0.072**	**0.218**	0.119

Generalization in Unseen Environments. Table 1 shows that LCVN-Uni demonstrates stronger navigation performance on the val unseen split, consistently outperforming LCVN-WM + LCVN-AC in SR while maintaining competitive ATE and RPE. This comparison reflects two practical system instantiations with different model scales and pretraining, so the observed advantage cannot be attributed to the unified architecture alone. Meanwhile, Table 2 shows that LCVN-WM achieves better long-horizon SSIM, PSNR, and LPIPS on val seen, whereas LCVN-Uni achieves better DreamSim. These findings highlight complementary strengths of the evaluated systems.

D Limitations

While LCVN provides a controlled testbed for studying language, imagination, and decision-making in embodied agents, several limitations remain. First, following prior goal-conditioned navigation work [74, 78, 73], we adopt an offline trajectory-prediction protocol, which enables reproducible comparison across heterogeneous real-robot data but does not capture closed-loop behaviors such as recovery from execution errors or interaction with dynamic agents; closed-loop validation on physical platforms is a natural complement. Second, all instructions in the LCVN dataset are written in English and follow three pre-defined styles (Concise, Intricate, Landmark-based); extending the benchmark to multilingual instructions and to noisier, more naturalistic phrasings that arise in real human–robot interaction is a promising direction. Third, our zero-shot evaluation relies primarily on TartanDrive's off-road driving scenes as the unseen split, which provides a strong distribution shift in terrain and embodiment but leaves additional axes of generalization open for future exploration.

Fourth, following the preprocessing protocol of [7, 78], we filter out backward movements and trajectories shorter than three steps to keep LCVN comparable with established goal-conditioned pipelines; this narrows the action distribution and removes some hard but realistic edge cases, and relaxing this filtering is a natural extension of the benchmark.

E Broader Impacts

LCVN targets language-conditioned visual navigation, which has natural applications in assistive robotics, service robots, and accessible human–robot interfaces, where users may prefer natural language over visual goals or low-level commands. We expect direct negative societal impacts to be limited: the world models studied in this work generate short-horizon egocentric navigation imagery rather than photorealistic media of identifiable individuals, and the underlying datasets are publicly released robotic trajectories without personally identifiable content. As with other navigation and generative-modeling research, downstream misuse in surveillance or in fabricating first-person footage is conceivable; we encourage subsequent work building on LCVN to follow responsible release practices and to consider provenance and access controls when deploying derived systems.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and Sec. 1 clearly state the three contributions (LCVN task formalization, LCVN Dataset, and the two paradigms LCVN-WM+LCVN-AC and LCVN-Uni), all of which are supported by Sec. 3–5 and the appendix.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: Limitations are discussed in Appendix D, covering the offline trajectory-prediction protocol, the language and instruction-style coverage of the LCVN dataset, and the scope of distribution shift in the unseen split.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [N/A]

Justification: The paper is empirical and does not contain formal theoretical results (theorems, lemmas, or convergence/generalization analyses). All equations in Sec. 4 and the appendix define model components, training objectives, and algorithmic procedures rather than claims requiring proof.

Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: Architectures are described in Sec. 4 and Figs. 2–3; training procedures are given as pseudo-code in Algorithms 1–3 (Appendix B); prompt design for LCVN-Uni is in Fig. A2; hyperparameters, optimizer, learning rate, batch size, and hardware are listed in Sec. 5.1 and Appendix C.2.

Guidelines:

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

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

   Answer: [Yes]

   Justification: The full LCVN Dataset (39,016 trajectories and 117,048 human-verified instructions across all five source datasets and all three instruction styles (Concise, Intricate, Landmark-based), with the train / val seen / val unseen / test splits used in the paper) is publicly released at https://huggingface.co/datasets/fly1113/LCVN, and the complete code for data preprocessing, training, and evaluation of LCVN-WM, LCVN-AC, and LCVN-Uni, together with the commands and environment needed to reproduce the main results, is released at https://github.com/UWMILab/LCVN. Architectures, pseudo-code (Algorithms 1–3), prompt design (Fig. A2), hyperparameters, optimizer, and hardware setup are additionally documented in Sec. 5.1 and Appendix B–C.2.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Data splits (train / val seen / val unseen / test with sizes 28,813 / 3,602 / 1,500 / 5,101) are given in Sec. 3.1; hyperparameters and implementation details are in Sec. 5.1 and Appendix C.2.

   Guidelines:

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

7. **Experiment statistical significance**

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

Answer: [No]

Justification: Due to the high computational cost of training diffusion world models and fine-tuning a 7B autoregressive backbone on 4×A100, results are reported from a single training run per configuration; we instead provide ablations along multiple axes (language, conditioning signals, instruction style, context size, encoding space, external data) to characterize variance qualitatively.

Guidelines:

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: All experiments are conducted on 4×NVIDIA A100 GPUs (80GB each), as stated in Sec. 5.1 and Appendix C.2. Per-step inference latency for each method is reported in Table A2.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research conforms to the NeurIPS Code of Ethics. All trajectory data is sourced from publicly released robotic navigation datasets, and human annotation was performed by an internal team for instruction verification.

Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: Both potential positive and negative societal impacts are discussed in Appendix E.

    Guidelines:

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

11. **Safeguards**

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

    Answer: [N/A]

    Justification: The released dataset is built on existing public navigation datasets (Go Stanford, ReCon, SCAND, HuRoN, TartanDrive), which do not contain personally identifiable information at high risk of misuse.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.

- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: All third-party assets used in this work are properly cited at first mention, including trajectory datasets (Go Stanford [37], ReCon [72], SCAND [41], HuRoN [38], TartanDrive [85], Ego4D [31]), pre-trained models (CLIP [71], Qwen-3.5-9B [83], GAIR Anole-7B [14]), and baseline implementations (DiT [69], NWM [7], Diamond [2], GNM [73], NoMaD [78]). We follow original licenses and intended research-use terms of these assets.

    Guidelines:

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

13. **New assets**

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

    Answer: [Yes]

    Justification: The LCVN Dataset is a new asset, documented in Sec. 3.1 and Appendix A (trajectory sources, preprocessing, seeded few-shot annotation pipeline, multi-style instruction examples in Table A1).

    Guidelines:

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

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

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

    Answer: [Yes]

Justification: As described in Sec. 3.1 and Appendix A, the LCVN dataset involves human annotators who authored seed instructions and verified/refined the Qwen-3.5-9B–generated instructions. Annotation was performed voluntarily by members of the research team (co-authors and lab collaborators), so no crowdsourcing platform or paid contract work was involved. The work did not include experimentation on human subjects.

Guidelines:

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

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

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

Answer: [N/A]

Justification: The work does not involve human subjects research beyond data annotation by an internal team using publicly released robotic trajectories; no personal or sensitive information about study participants was collected.

Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [Yes]

Justification: Qwen-3.5-9B is used as a core component of the dataset construction pipeline: it segments trajectories into semantically coherent sub-scenes (Sec. 3.1) and generates candidate instructions in three styles under seeded few-shot prompting, which are subsequently verified and refined by human annotators (Appendix A).

Guidelines:

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