Planning in 8 Tokens: A Compact Discrete Tokenizer for Latent World Model

Dongwon Kim[1] Gawon Seo[2] Jinsung Lee[2] Minsu Cho[2,3] Suha Kwak[2]
[1]KAIST [2]POSTECH [3]RLWRLD
{kdwon}@kaist.ac.kr {gawon, jinsunglee, mscho, suha.kwak}@postech.ac.kr

Abstract

World models provide a powerful framework for simulating environment dynamics conditioned on actions or instructions, enabling downstream tasks such as action planning or policy learning. Recent approaches leverage world models as learned simulators, but its application to decision-time planning remains computationally prohibitive for real-time control. A key bottleneck lies in latent representations: conventional tokenizers encode each observation into hundreds of tokens, making planning both slow and resource-intensive. To address this, we propose CompACT, a discrete tokenizer that compresses each observation into as few as 8 tokens, drastically reducing computational cost while preserving essential information for planning. An action-conditioned world model that occupies CompACT tokenizer achieves competitive planning performance with orders-of-magnitude faster planning, offering a practical step toward real-world deployment of world models.

1. Introduction

Humans navigate the world not through pixel-perfect recall of their surroundings, but rather through compact mental representations that capture only the information necessary for decision-making [23, 28]. This internal model—an imprecise but efficient abstraction of reality—reduces the complexity of sensory input into a representation optimized for action and planning. In the context of artificial intelligence and reinforcement learning (RL), this concept manifests as the *world model* [28], a neural network that captures environment dynamics to enable planning [3, 32, 49, 77] and policy learning [1, 29–31, 69].

World models have emerged as a promising solution to the sample inefficiency of RL. Traditional model-free RL methods require millions of interactions with the environment to learn effective policies, making them impractical for real-world applications where data collection is expensive or risky. By learning to predict future states, world models enable agents to simulate experiences internally, reducing the need for real environment interactions. Furthermore, these models themselves can be used for planning without additional learning of policy [3, 77] through model-predictive control (MPC) [15, 68].

Recent advances in world modeling have been driven by the rapid progress in generative models, particularly in image and video generation [8, 19, 56]. These models can generate photorealistic images or videos conditioned on language instructions [17, 69] or actions [1, 3, 69, 77, 78], suggesting an implicit understanding of world's underlying dynamics.

However, there exists a critical gap between these generative approaches and their application to planning. These models are designed for photorealistic image generation, requiring them to capture extensive perceptual detail such as textures, lighting, and shadows. This necessitates encoding single images into hundreds of latent tokens, which sharply increases computational cost. Since most world models in the literature adopt attention-based architectures [54], this burden grows quadratically, making planning especially expensive. As a result, current world models remain impractical for real-world control: for example, the state-of-the-art navigation world models (NWM) [3] require up to 3 minutes of computation per episode for planning,[1] making them unsuitable for applications demanding real-time responsiveness. This motivates us to explore an alternative design philosophy: *what if we prioritize extreme compression over perfect reconstruction?* Rather than seeking the conventional path of increasing token count for higher fidelity representations, we hypothesize that aggressive compression might actually be beneficial—forcing the world model to learn more abstract, action-relevant representations rather than preserving every perceptual detail. To test this hypothesis, we push compression to its extreme limit and investigate whether such radical reduction can still support effective planning.

We propose CompACT, a compact tokenizer that encodes each image as few as 8 tokens—approximately 128 bits per image (8 tokens of 16 bits each). This represents an extreme compression ratio compared to existing approaches. For instance, the SD-VAE tokenizer [56] used in NWM [3]

[1]Measured using a single RTX 6000 ADA GPU.

requires 784 tokens to represent the same image. Beyond the reduction in token count, our tokenizer further distinguishes itself by employing a discrete latent space, enabling much faster future-state prediction: each token is unmasked only once [8], rather than being processed through hundreds of iterative denoising steps typically required in diffusion models utilizing continuous latent space [35]. By training world models in this compact latent space, we can achieve order-of-magnitude reductions in rollout latency.

Compressing each image to just 128 bits creates an irreducible information bottleneck—the question is not whether to lose information, but which information to preserve. Planning requires low-frequency features such as high-level semantics and spatial relationships, rather than high-frequency perceptual details like textures and lighting. Our approach separates these two aspects: only planning-critical semantics are *preserved* in compact tokens, while perceptual details are *synthesized* when pixel-level outputs are needed during decoding.

The key design choice enabling such selective preservation of semantic information is our use of a *frozen* pretrained vision encoder [59] as the foundation of our tokenizer. Conventional tokenizers train encoders end-to-end for reconstruction, prioritizing perceptual fidelity. In contrast, we leverage the rich semantic representations already captured by vision foundation models. Our compact latent tokens act as learnable queries that attend to these frozen representations via a cross-attention-based resampling module. Crucially, because vision foundation models already abstract away low-level reconstruction details—focusing instead on semantic understanding—our resampling process can only distill planning-critical semantic information. This design inherently ensures the tokenizer preserves object-level semantics and spatial relationships over photorealistic details.

Complementing this semantic encoding strategy, our second key contribution is a generative decoding approach: rather than attempting direct pixel reconstruction from 16 or 8 tokens, our decoder learns to unmask a latent representation capturing perceptual details from a pretrained target tokenizer that uses hundreds of tokens per image (specifically, the VQGAN tokenizer from MaskGIT [8]), using our compact tokens as conditioning. While our compact latent tokens capture only high-level semantic features, the generative decoding process synthesizes fine-grained details that are consistent with these semantics. This formulation transforms an intractable decompression problem into a tractable conditional generation task.

To validate the effectiveness of the proposed approach, we train action-conditioned world models on the latent space of CompACT for both navigation and robot manipulation tasks. Such action-conditioned world models have a unique strength in that they can serve as general-purpose planners via MPC, but the prohibitive computational burden required for rollouts has remained as a bottleneck. On navigation planning in RECON [58], an action-conditioned world model trained with CompACT achieves comparable accuracy to one using 784 continuous tokens while delivering approximately 40× speedup in planning latency. Furthermore, our 8-token model outperforms previous tokenizer with 64 tokens, validating that carefully designed extreme compression can yield both computational efficiency and superior planning performance. To further validate the efficacy of compact latent tokens learned by CompACT, we conduct action-conditioned video prediction experiments on RoboNet [14]. On RoboNet, CompACT latent tokens enable accurate action regression comparable to previous tokenizers using 16× more tokens, and maintain strong action consistency in generated videos, confirming that the learned representations preserve action-relevant information critical for accurate planning.

2. Related Work

2.1. Image tokenization

Image tokenization has played a crucial role in visual generation. Encoding raw image pixels into compressed latent representations alleviates the difficulty of directly modeling distributions in high-dimensional continuous spaces. By discretizing these latents, the model can efficiently produce and sample categorical distributions for individual tokens. Autoencoder-based architectures such as VQ-VAE [64], VQ-GAN [19], ViT-VQGAN [56], and Efficient-VQGAN [6] employ vector quantization to form discrete latent spaces. Numerous enhancements have been proposed to improve reconstruction quality, including perceptual losses [74], adversarial losses [26], transformer-based designs, residual quantization [43], lookup-free quantization [72], and finite scalar quantization (FSQ) [48].

A common limitation of the aforementioned approaches is their reliance on 2D patch-grid latent representations. This design fixes the number of tokens according to the image resolution (H, W) and prevents its adaptive adjustment based on image complexity. To overcome this, recent works have explored 1D tokenization [2, 40, 50, 73], which does not explicitly preserve spatial structure. For example, TiTok [73] employs learned register tokens to capture image content in a compact sequence via a ViT encoder, enabling efficient representation learning. FlexTok [2] allows flexible token lengths ranging from 1 to 256 tokens where later tokens capture progressively finer details. However, existing image tokenizers are designed for the image generation, prioritizing high-frequency details and high-fidelity reconstruction. In contrast, our tokenizer aggressively compresses scenes by preserving only essential visual information. This compact representation enables agents to find optimal plans faster.

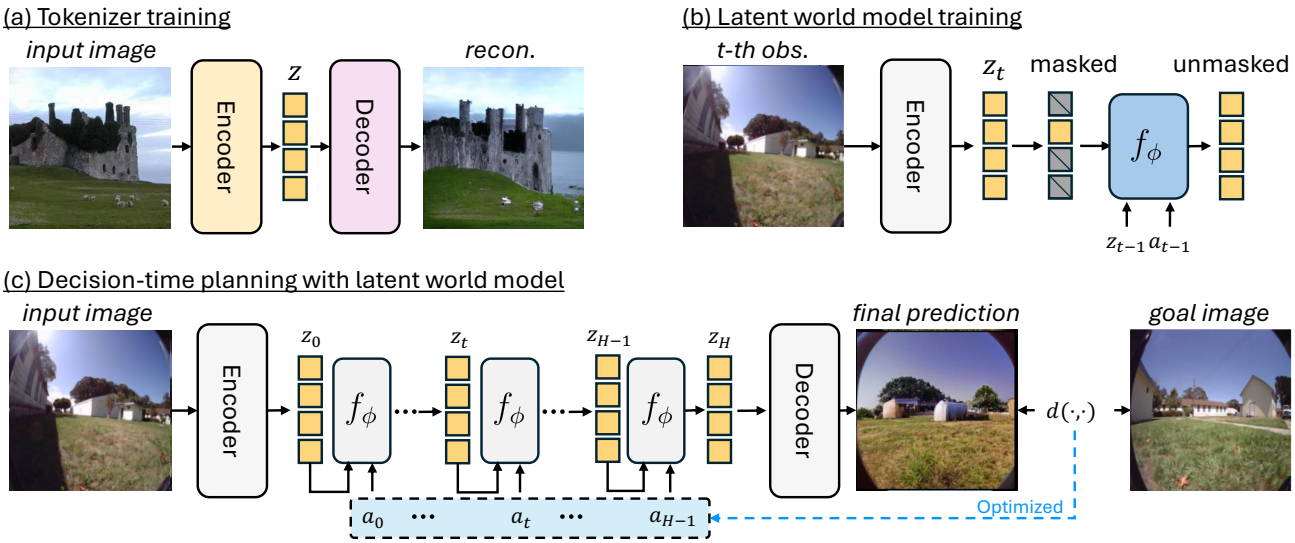

Figure 1. **Overview of the proposed latent world model formulation (Sec. 3.1).** (a) An image tokenizer is first trained with a reconstruction objective to map an input image into compact latent tokens $\boldsymbol{z}$. (Fig. 2 and Sec. 3.2). (b) Using the learned tokenizer, latent world model $f_\phi(\boldsymbol{z}_t, \boldsymbol{a}_t)$ is trained to model the conditional distribution of the future state $p_\phi(\boldsymbol{z}_{t+1}|\boldsymbol{z}_t, \boldsymbol{a}_t)$, where we adopt masked generative modeling (Sec. 3.3). (c) At test time, the learned latent world model is used for *decision-time planning*: An optimization procedure (e.g., MPC with CEM) searches over actions $\boldsymbol{a}_{0:H-1}$ to minimize the distance between the predicted final state and a goal image.

2.2. Masked generative model

Masked image generative models [8, 9, 22, 24, 44, 45, 67] leverage bidirectional attention mechanisms to reconstruct masked tokens during generation. Unlike traditional autoregressive models [11, 62] predicting tokens one-by-one, these architectures [72, 76] can sample multiple tokens within a single step, thereby reducing the number of steps needed for full image generation. Notably, MaskGIT [8] and MAR [45] have demonstrated that such designs enable both rapid and high-quality image synthesis. In this work, we focus on the tokenization stage and adopt the widely used non-autoregressive sampling approach from MaskGIT [8] for generating token sequences that are subsequently decoded into 2D discrete latent tokens that serve as the input to the VQGAN decoder in our pipeline.

2.3. Planning via World Models

World models [28] serve as internal representations that encode environmental dynamics, enabling agents to mentally simulate future states before acting. By predicting future observations from current states and actions, these models facilitate planning across diverse domains including robotics [21, 47, 69], autonomous driving [25, 36, 75], gaming [1, 5, 63], and navigation [3, 42, 52, 70]. Existing approaches can be broadly categorized into two paradigms based on their planning mechanisms. One line of approaches is action-conditioned world models which employ test-time optimization, where methods like TDMPC2 [32] and NWM [3] integrate learned dynamics with model-predictive control (MPC) [15] to iteratively refine action sequences toward specified goals. Another line of approaches adopts hierarchical planning through subgoal generation, where sparse intermediate visual states are first generated to bridge current observations to goals, followed by Inverse Dynamics Models to extract executable actions. UniPi [18] exemplifies this strategy through conditional video generation guided by textual goals, and AVDC [41] uses language-conditioned prediction with optical flow for action estimation. While these existing approaches face significant computational challenges in real-time settings, especially with large models like diffusion-based video generation models [5, 18, 36, 69]. This constraint has motivated the development of latent-space alternatives to enhance efficiency. In this work, we aim to build a world model within the extremely compact latent space instead of the computationally expensive raw pixel space, enabling more precise planning and control.

3. Method

3.1. Latent generative model as world model

In this section, we first describe how a world model can be formulated as a latent generative model. The overall formulation is depicted in Fig. 1. We consider the standard world model setting where the objective is to predict future observations given current state and action. Formally, we denote observations (e.g., video frames) as $\boldsymbol{O} = [\boldsymbol{o}_0, \boldsymbol{o}_1, \ldots, \boldsymbol{o}_T] \in \mathbb{R}^{T\times H\times W\times 3}$ and actions as

$\boldsymbol{A} = [\boldsymbol{a}_0, \boldsymbol{a}_1, \ldots, \boldsymbol{a}_T] \in \mathbb{R}^{T\times 3}$.[2] The world model $f_\theta : \mathbb{R}^{H\times W\times 3} \times \mathbb{R}^3 \to \mathcal{P}(\mathbb{R}^{H\times W\times 3})$ can be formulated as:

$$f_\theta : (\boldsymbol{o}_t, \boldsymbol{a}_t) \mapsto p_\theta(\boldsymbol{o}_{t+1}|\boldsymbol{o}_t, \boldsymbol{a}_t). \tag{1}$$

For simplicity, we omit the temporal context window in our notation; in practice, the model conditions on a history of τ observations and actions.

Because real-world dynamics are inherently uncertain and only partially observable, a world model should produce a stochastic distribution over future states rather than a deterministic prediction. Such a stochastic formulation of the world model can be naturally implemented using generative modeling, where the generator is conditioned on past observations $\boldsymbol{o}_t$ and action $\boldsymbol{a}_t$. Direct generative modeling in pixel space is computationally prohibitive due to the high dimensionality of visual observations. Instead, the world model f_θ can be formulated to operate on low-dimensional latent tokens $\boldsymbol{z} \in \mathbb{R}^{N\times D}$ [3]. These latent tokens are obtained via an image tokenizer comprising an encoder $\mathcal{E} : \mathbb{R}^{H\times W\times 3} \to \mathbb{R}^{N\times D}$ and decoder $\mathcal{D} : \mathbb{R}^{N\times D} \to \mathbb{R}^{H\times W\times 3}$, trained with a reconstruction objective: $\mathcal{L}_{\text{recon}} = ||\boldsymbol{o} - \mathcal{D}(\mathcal{E}(\boldsymbol{o}))||_2^2$ (Fig. 1(a)). Extending Eq. (1), a latent world model $f_\phi : \mathbb{R}^{N\times D}\times\mathbb{R}^3 \to \mathcal{P}(\mathbb{R}^{N\times D})$ can be described as

$$f_\phi : (\boldsymbol{z}_t, \boldsymbol{a}_t) \mapsto p_\phi(\boldsymbol{z}_{t+1}|\boldsymbol{z}_t, \boldsymbol{a}_t), \tag{2}$$

where $\boldsymbol{z}_t = \mathcal{E}(\boldsymbol{o}_t)$. Here, the token count N directly determines computational complexity: for attention-based architectures [54] commonly used in generative models, cost scales quadratically with N. By keeping N small, the latent world model formulation alleviates this quadratic bottleneck and enables efficient decision-time planning.

Once the latent world model f_θ is trained, we can use it to find a sequence of actions $\{\boldsymbol{a}_t\}$ that drives the transition from the initial observation $\boldsymbol{o}_0$ to the goal observation $\boldsymbol{o}_{\text{goal}}$, as illustrated in Fig. 1(c). We first compute $\boldsymbol{z}_0 = \mathcal{E}(\boldsymbol{o}_0)$, and initialize a candidate action sequence $\mathbf{a} = [\boldsymbol{a}_0, \boldsymbol{a}_1, \ldots, \boldsymbol{a}_{H-1}]$. Then, we obtain a sequence of latent tokens $\{\boldsymbol{z}_t\}$ by rolling out the trained world model to predict future states over the planning horizon H:

$$\boldsymbol{z}_{t+1} \sim f_\phi(\boldsymbol{z}_t, \boldsymbol{a}_t),\ t \in \{0, \cdots, H-1\}. \tag{3}$$

After the rollout reaches the planning horizon (i.e., $\boldsymbol{z}_H$ is sampled), the candidate action sequence $\mathbf{a}$ is evaluated using a cost function that measures the distance between the final predicted observation and the goal: $C(\mathbf{a}) = d(\hat{\boldsymbol{o}}_H, \boldsymbol{o}_{\text{goal}})$, where $\hat{\boldsymbol{o}}_H = \mathcal{D}(\boldsymbol{z}_H)$, $\hat{\boldsymbol{o}}_{\text{goal}} = \mathcal{D}(\boldsymbol{z}_{\text{goal}})$, and $d(\cdot,\cdot)$ is a distance measure (e.g., LPIPS [38]).[3] The optimal action sequence is then obtained via solving $\mathbf{a}^* = \arg\min_{\mathbf{a}} C(\mathbf{a})$, where the optimization can be performed using sampling-based methods [13, 15, 68] or gradient descent.

[2] In navigation settings, actions are 3-dimensional, representing changes in x-axis, y-axis, and yaw. The formulation generalizes to different action dimensions (e.g., 5-dimensional actions of a robot arm).

[3] We can also define the cost function as $d(\boldsymbol{z}_H, \boldsymbol{z}_{\text{goal}})$ in the latent space, which enables the faster planning since we can skip the decoding (Tab. 5).

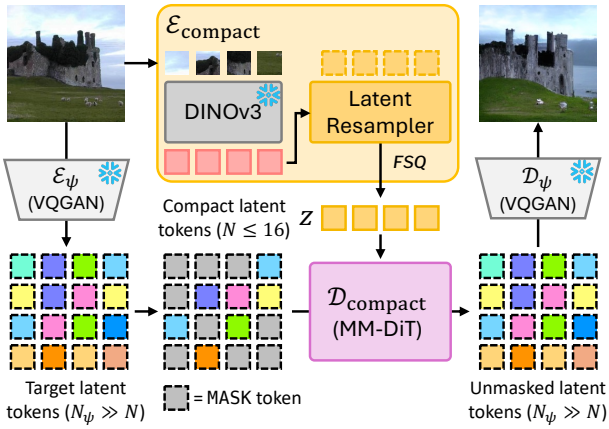

Figure 2. **A tokenizer architecture detail.** During training, only the latent resampler and $\mathcal{D}_{\text{compact}}$ are updated. $\mathcal{E}_\psi$ produces masked target tokens (training only), while $\mathcal{D}_\psi$ is used only during inference for pixel level reconstruction.

3.2. CompACT tokenizer

The computation bottleneck in world model planning stems from the latent token count N: conventional tokenizers typically encode images with hundreds of tokens, which slows down their sampling during autoregressive rollout. We introduce CompACT, a compact tokenizer $\mathcal{D}_{\text{compact}} \circ \mathcal{E}_{\text{compact}}$ that encodes each image into just 16 or 8 discrete tokens and avoids iterative denoising by using a discrete latent space (Fig. 2). Despite this extreme compression, CompACT still preserves the sufficient information for planning (Sec. 4).

3.2.1. Semantic encoding via frozen features

The key design principle of our tokenizer is to preserve only planning-critical semantic information while discarding reconstruction-oriented high-frequency features. To achieve this, we build our encoder $\mathcal{E}_{\text{compact}}$ around a *frozen* pretrained vision encoder—specifically, DINOv3 [59]—which already abstracts away low-level visual details in favor of semantic understanding. The encoder $\mathcal{E}_{\text{compact}} : \mathbb{R}^{H\times W\times 3} \to \{1, \ldots, K\}^N$ maps an input image $\boldsymbol{o}$ into a sequence of $N (N \le 16)$ discrete tokens $\boldsymbol{z}$, each selected from a vocabulary of size K. The encoder architecture consists of three components: (1) a frozen DINOv3 model that extracts semantic patch representations, (2) a latent resampler with learnable query tokens, and (3) a finite scalar quantization layer [48].

Specifically, the input image is patchified and encoded by the frozen DINOv3 model to obtain semantic representations. The initial latent tokens $\boldsymbol{z}^0 \in \mathbb{R}^{N\times D}$ then act as learnable queries in a transformer decoder-based latent resampler [65]. In each decoder block, these latent tokens

attend to the DINOv3 output patch tokens via cross-attention layers, effectively distilling high-level semantic cues from the pretrained representations. Because the vision foundation model has already abstracted away textures, lighting, and other low-level details, the cross-attention mechanism can selectively focus on semantic information—object identities, spatial layouts, and scene structure—that remains in the frozen features. The output of the latent resampler is then discretized using finite scalar quantization, yielding discrete latent tokens $\boldsymbol{z} \in 1, \ldots, K^N$. While such extreme compression inevitably discards fine-grained visual details, we hypothesize that these details are largely irrelevant for planning tasks, where object-level semantics and spatial relationships dominate decision-making.

3.2.2. Generative decoding

Direct pixel reconstruction from $N \leq 16$ tokens is an ill-posed problem—the information bottleneck prevents the deterministic recovery of perceptual details, since diverse pixel-space manifestations can arise from identical semantic features. To address this, we propose a generative decoding strategy that introduces an intermediate representation. Our decoder $\mathcal{D}_{\text{compact}} : \{1, \ldots, K\}^N \to \{1, \ldots, K_\psi\}^{N_\psi}$ learns to generate latent tokens from a pretrained tokenizer $\mathcal{D}_\psi \circ \mathcal{E}_\psi$ [8], using our compact tokens $\boldsymbol{z}$ as a condition. We refer to this pretrained tokenizer as the *target tokenizer* because its tokens serve as intermediate targets that bridge our semantic representation to pixel space. Specifically, we employ the VQGAN from MaskGIT [8], which encodes images into hundreds of tokens ($N_\psi \gg N$, typically $N_\psi = 196$ for 224×224 images) capturing perceptual details omitted in our compact tokens. This transforms the intractable decompression problem into a conditional generation task.

Specifically, we first convert an image $\boldsymbol{o}$ into target tokens $\boldsymbol{z}^\psi = \mathcal{E}_\psi(\boldsymbol{o}) \in \{1, \ldots K_\psi\}^{N_\psi}$ using the pretrained tokenizer encoder, where $N_\psi \gg N$ (typically $N_\psi = 196$ for 224×224 images). We then employ masked generative modeling [8, 72] to learn the mapping from $\boldsymbol{z}$ to $\boldsymbol{z}^\psi$, which offers significantly faster sampling than autoregressive models [4, 61]. During training, a random subset of the target tokens $\boldsymbol{z}^\psi$ is masked, and the decoder learns to recover them using the compact tokens $\boldsymbol{z}$ and the remaining unmasked tokens. The tokenizer training objective is defined to minimize the negative log-likelihood of the masked tokens $\boldsymbol{z}^\psi$:

$$\mathcal{L}_{\text{tok}} = -\mathop{\mathbb{E}}_{\boldsymbol{z}^\psi} \big[\log p(\boldsymbol{z}^\psi | \boldsymbol{z}, M(\boldsymbol{z}^\psi)) \big], \tag{4}$$

where $M(\cdot)$ represents the random masking. CompACT is trained solely with the unmasking objective in latent space without pixel-level reconstruction, where the weights of the target tokenizers are not updated. During inference, $\mathcal{D}_{\text{compact}}$ begins with a fully masked sequence and iteratively unmasks them following the sampling scheme based on its prediction confidence [8]. The compact tokens $\boldsymbol{z}$ provide high-level semantic guidance throughout this process, while the generative model synthesizes plausible visual details consistent with these semantics. The final reconstruction is obtained through the target decoder: $\hat{\boldsymbol{o}} = (\mathcal{D}_\psi \circ \mathcal{D}_{\text{compact}} \circ \mathcal{E}_{\text{compact}})(\boldsymbol{o})$.

In a nutshell, our CompACT tokenizer achieves extreme compression by preserving only high-level semantics in $N (N \leq 16)$ discrete tokens, then using these as conditioning for a generative decoder that synthesizes plausible high-frequency details. This design aligns with our core hypothesis that effective planning requires not photorealistic world models, but compact representations of decision-critical information.

3.3. World model in CompACT latent space

With our CompACT tokenizer defined, we can now train the world model formulated in Eq. (2) directly in the N-token discrete latent space ($N \leq 16$), as shown in Fig. 1(b). Given a dataset of observations and action sequences, we first encode all observations into compact latent tokens using CompACT tokenizer: $\boldsymbol{z}_t = \mathcal{E}_{\text{compact}}(\boldsymbol{o}_t)$. Similar to generative decoding, we use the masked generative modeling to train the world model f_ϕ. The training objective is given by

$$\mathcal{L}_{\text{world}} = -\mathop{\mathbb{E}}_{z_t, a_t, z_{t+1}} \big[\log p(\boldsymbol{z}_{t+1} | \boldsymbol{z}_t, \boldsymbol{a}_t, M(\boldsymbol{z}_{t+1})) \big]. \tag{5}$$

The key advantage of this formulation is computational efficiency during planning. During model-predictive control, it can now perform rollouts using only $N (N \leq 16)$ tokens per timestep, enabling planning latency that was previously intractable with hundred-length tokens.

Since the specific choice of world model architecture is orthogonal to our tokenizer design, any model capable of modeling discrete sequence distributions can be employed. We explore two frameworks for learning the conditional distribution $p(\boldsymbol{z}_{t+1} | \boldsymbol{z}_t, \boldsymbol{a}_t)$. For navigation tasks, we follow an autoregressive framework following NWM [3]: at each step, the model predicts $\boldsymbol{z}_{t+1}$ conditioned on a fixed-length history window of latents $\{\boldsymbol{z}_{t-\tau}, \ldots, \boldsymbol{z}_t\}$ and actions $\{\boldsymbol{a}_{t-\tau}, \ldots, \boldsymbol{a}_t\}$, implemented using a DiT-based architecture [54]. To improve action conditioning, we randomly mask latent tokens in the history window during training. For robotic manipulation on RoboNet [14], we employ a block-causal transformer that models multiple future frames simultaneously, predicting $\{\boldsymbol{z}_{t+1}, \ldots, \boldsymbol{z}_{t+K}\}$ in parallel while maintaining causal dependencies between frames.

Both training schemes can be understood as discrete variants of the diffusion forcing [10]: the navigation model learns to condition on partially masked context, while the parallel generation naturally implements diffusion forcing as frames at different unmasking stages provide varying levels of noisy conditioning. This robust training improves planning accuracy without additional cost (ablation in Table 5; see the supplement for implementation details).

4. Experiment

4.1. Experimental Settings

We evaluate CompACT across two key aspects: (1) **tokenization quality** through reconstruction metrics, and (2) **planning effectiveness** through action-conditioned world models in navigation and manipulation tasks. This dual evaluation validates our hypothesis that extreme compression preserves planning-critical information while enabling efficient decision-time planning.

Task Conductive. We evaluate CompACT on the following tasks: (1) Image reconstruction: Reconstructing original images from compressed latent tokens. (2) Goal-conditioned visual navigation: Given a context image and a navigation goal image, planning optimal paths by predicting future observations conditioned on actions. (3) Action-conditioned video prediction: Generating future video frames conditioned on current visual input and actions.

Dataset. We train CompACT on ImageNet-1K [16]. World models are trained on three navigation datasets following NWM [3]: RECON [58], SCAND [39], and HuRoN [34]. For manipulation, we use RoboNet [14], which contains diverse robot interaction data across multiple environments.

Tokenizer baselines. We compare CompACT against two baseline tokenizers: (1) SD-VAE [56]: A continuous latent space tokenizer requiring 784 tokens per image (28×28 spatial grid), representing the conventional approach used in state-of-the-art world models like NWM [3]. This establishes the computational cost we aim to reduce. (2) FlexTok [2]: A recent discrete tokenizer supporting flexible token lengths (1-256 tokens). We evaluate it at 16 and 64 tokens to directly compare against CompACT's compression levels.

Evaluation. We employ task-appropriate metrics:

- **Reconstruction quality**: reconstruction FID (rFID) [53] and Inception Score (IS) [57] on ImageNet validation set.
- **Planning accuracy**: Absolute Trajectory Error (ATE) and Relative Pose Error (RPE) for navigation tasks, measuring how closely planned trajectories match ground truth.
- **Action-relevancy**: Inverse Dynamics Model (IDM) performance using L1 error and coefficient of determination on predicted end-effector positions, validating that compact tokens preserve action-critical information.
- **Action-conditioned video prediction**: Action Prediction Error (APE), measured as the L1 error between the conditioning action and the action predicted by IDM from generated frames. This evaluates whether generated videos accurately reflect the dynamics induced by the actions.
- **Computational efficiency**: Planning latency during model-predictive control.

Implementation Details. We use DINOv3-B [59] for pretrained vision encoder in $\mathcal{E}_{\text{compact}}$. For the generative decoder, we use VQGAN from MaskGIT [8] as the target tokenizer $\mathcal{D}_\psi \circ \mathcal{E}_\psi$. For detailed hyperparameters for training and model architecture, we refer readers to the supplement.

Table 1. **Reconstruction performance of CompACT on ImageNet validation split.** Metrics are computed using open-sourced checkpoints. rFID is measured using clean-fid [53]. †: Measured using 16 tokens.

Model	Type	#tok	rFID ↓	IS ↑
SD-VAE [56]	cont.	1024	0.64	223.8
MaskGIT-VQGAN [8]	disc.	256	1.83	186.7
TA-TiTok-VQ [40]	disc.	32	3.95	219.6
TA-TiTok-KL [40]	cont.	32	1.93	222.0
FlexTok [2]	disc.	1-256	5.60†	114.9
CompACT	disc.	16	2.40	209.0
CompACT	disc.	8	3.21	207.5

Table 2. **Ablation on CompACT tokenizer.** rFID is measured on ImageNet [16] validation split using clean-fid [53].

Configuration		rFID↓	#tok
Target tokenizer ($\mathcal{D}_\psi \circ \mathcal{E}_\psi$) [8]		1.83	256
DINOv3-B (frozen) + latent resampler		2.40	16
$\mathcal{D}_{\text{compact}}$	w/o generative decoding	28.80	16
$\mathcal{E}_{\text{compact}}$	ViT-B (trained from scratch) + [REG]	7.28	16
	DINOv3-B (finetuned) + [REG]	4.51	16
	DINOv3-B (finetuned) + latent resampler	5.22	16

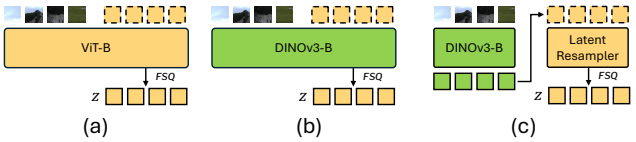

Figure 3. **CompACT encoder $\mathcal{E}_{\text{compact}}$ architecture variation.** (a) ViT (scratch)+ [REG]: Initial latent tokens are concatenated to the input patch tokens. This design follows previous transformer-based image tokenizers [2, 71, 73]. (b) DINOv3 [59] + [REG]: Similar to (a), but encoder is initialized with Dinov3. (c) DINOv3 [59] + latent resampler: latent resampler and Dinov3 initialized encoder. Dino and ViT are updated during training in these variants.

4.2. Tokenizer evaluation and ablations

Reconstruction performance. Table 1 presents the reconstruction quality of each tokenizer measured by rFID and IS. MaskGIT-VQGAN [8], which is used as a target tokenizer $\mathcal{D}_\psi \circ \mathcal{E}_\psi$ in CompACT, serves as a baseline for reconstruction fidelity since CompACT relies on $\mathcal{D}_\psi$ for final pixel reconstruction. The results demonstrate that CompACT can attain reconstruction performance comparable to recent state-of-the-art tokenizers while achieving extreme compression rate. Interestingly, CompACT outperforms MaskGIT-VQGAN [8] in IS, suggesting that our semantic encoder better preserves perceptually-relevant features that may be overlooked by VQGAN's pixel-focused reconstruction objective.

Ablation of encoder design choices. We report ablation studies on our final CompACT tokenizer design choices in

Figure 4. **Attention visualization for compact latent token in latent resampler.** Brighter the color, higher the attention score.

Table 2. We compare three encoder architecture variants illustrated in Fig. 3. The results demonstrate that leveraging frozen features from the semantic encoder is a key design choice enabling extremely compact tokenization while maintaining strong reconstruction quality. Notably, full finetuning of DINOv3-B results in significantly worse rFID of 5.22. We conjecture that finetuning the vision foundation model shifts its representations toward reconstruction-oriented features, causing the compact latent tokens to lose the high-level semantic information that is crucial for our generative decoding approach. Without rich semantic conditioning, the generative decoder cannot synthesize plausible fine-grained details, leading to degraded reconstruction quality.

Ablation of generative decoding. Table 2 also shows the ablation studies for the generative decoding. To check the effectiveness of the generative decoding, we replaced the $\mathcal{D}_{\text{compact}}$ with the single-step feedforward decoder, which leads to the severe degradation in reconstruction quality. This results highlights the $\mathcal{D}_{\text{compact}}$'s crucial role in synthesizing fine-grained features, as compact latent token only preserves high-level semantic features.

4.3. Characterizing CompACT latent tokens

CompACT tokens capture modular scene elements. Fig. 4 visualizes the attention maps from the latent resampler across ImageNet, RECON, and RoboNet. We observe that each compact latent token attends to coherent, semantically meaningful regions within the image, effectively capturing modular object-level elements. This compositional structure emerges naturally from frozen DINO features used for latent resampling, as they already encode object-centric representations. Specifically, in ImageNet, tokens attend to distinct objects (animals); in RECON, they focus on structural elements like buildings; in RoboNet, they isolate the end-effector and manipulation targets. Rather than distributing information uniformly across tokens, CompACT learns to allocate each token to semantically coherent scene components without explicit supervision.

Modular latents benefit planning. We now validate whether this modular structure translates to improved planning performance. Since planning requires understanding how actions induce state transitions, we use an Inverse Dynamics Model (IDM) as a proxy to evaluate whether our compact tokens preserve dynamics-relevant information: if we can train an IDM that accurately predicts actions from consecutive frames on latent representations, it indicates that the latents capture the essential state changes for control tasks. For details of IDM implementation, please refer to the supplementary material. As shown in Table 3, the IDM trained on CompACT latents achieves superior performance compared to MaskGIT-VQGAN despite using 16× fewer tokens. For this experiment, we finetune CompACT (pretrained on ImageNet) on RoboNet to adapt the representations to the manipulation domain. This performance gap reveals a fundamental difference in token allocation: CompACT's modular tokens naturally capture dynamic objects—the end-effector and manipulation targets—whose state transitions encode action information. By allocating tokens to semantically coherent objects rather than fixed spatial regions, our compact representation better captures the state changes induced by actions, benefiting downstream planning tasks.

Table 3. **Performance of Inverse Dynamics Model (IDM) trained with different tokenizers on RoboNet [14].** L1 error and R^2 are measured between ground truth and predicted end effector position.

Tokenizer	#tok	L1 err↓	R^2 ↑
Target tokenizer ($\mathcal{D}_\psi \circ \mathcal{E}_\psi$) [8]	256	0.093	0.684
CompACT	16	0.091	0.716

Table 4. **Planning performance of NWM on RECON benchmark with different tokenizers.** Latency (sec) represents single trajectory optimization time using a single RTX 6000 ADA GPU.

Tokenizer	#tok	RECON		SCAND		Latency↓
		ATE↓	RPE↓	ATE↓	RPE↓	
SD-VAE [56]	784	**1.262**	**0.354**	**1.065**	**0.291**	178.78
FlexTok [2]	64	1.484	0.400	1.578	0.378	16.68
FlexTok [2]	16	1.625	0.446	1.503	0.362	14.48
CompACT	16	1.330	0.390	1.358	0.336	5.78
CompACT	8	1.373	0.401	1.391	0.346	**4.83**

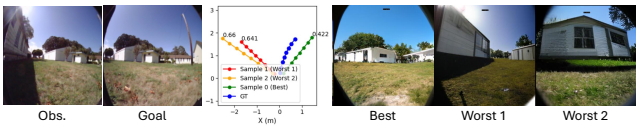

Figure 5. **Qualitative results of planning with the proposed CompACT.** The final rollout corresponding to the simulated trajectory with the minimum cost is highlighted in red.

4.4. Planning in CompACT latent space

Planning performance. Table 4 presents planning results for goal-conditioned visual navigation on the RECON dataset. Our CompACT achieves approximately *40× reduc-*

Table 5. **Effect of design choices in terms of planning accuracy on RECON.** (*Left*): Effect of the history masking in the world model f_θ (Sec. 3.3). (*Middle*): Comparison between the different cost function (Sec. 3.1). (*Right*): Effect of the freezing vision encoder during tokenizer training (Sec 3.2.1).

Hist. mask	ATE
✓	1.330
	1.480

Cost	ATE	Latency
Pixel	1.330	5.78sec
Latent	1.379	2.15sec

Frozen	ATE
✓	1.330
	1.472

tion in planning latency while maintaining comparable planning accuracy to the SD-VAE baseline that uses 784 tokens. Notably, NWM trained with CompACT (using only 16 or 8 tokens) outperforms the FlexTok-based model at both 16 and 64 token configurations. This performance advantage stems from our encoder's design: $\mathcal{E}_{\text{compact}}$ leverages frozen semantic features from pretrained vision foundation models for latent token resampling, ensuring that semantically meaningful features—spatial relationships, object configurations, and scene structure—are prioritized over reconstruction-oriented details like textures or lighting (Fig. 4). This semantic prioritization enables the world model to focus on action-relevant state transitions, as further validated by our inverse dynamics modeling results (Tab. 3).

Qualitative examples. Fig. 14 presents qualitative planning results with CompACT. While fine-grained visual details such as textures and shadows are synthesized rather than reconstructed, the rollouts accurately preserve planning-critical information needed for effective goal-reaching: spatial layout and object positions.

In-depth analysis on the effect of history masking, tokenizer, and cost function. Table 5 analyzes three key design choices that contribute to our method's effectiveness. (*Left*) *History masking*: Results demonstrate that incorporating history masking during world model training improves planning accuracy, validating that masking encourages robust temporal dependency learning. (*Middle*) *Cost function*: We compare planning accuracy when the cost is computed in pixel space (LPIPS) versus latent space (L1 distance between $\boldsymbol{z}$). While LPIPS achieves marginally better planning accuracy, computing distances in latent token space offers substantial computational benefits—achieving nearly 80× speedup in planning time compared to SD-VAE-based planning (Tab. 4). (*Right*) *Frozen encoder*: Similar to the reconstruction quality degradation in Tab. 2, updating the vision encoder during tokenizer training leads to degraded planning performance as well. Fine-tuning the vision foundation model shifts its representations toward reconstruction objectives, causing compact tokens to lose high-level semantic information essential for planning.

Action conditioned video prediction. To further validate that CompACT's modular latent representation preserves action-relevant information, we evaluate action-conditioned video generation on RoboNet [14]. Table 6 shows that CompACT achieves 3× lower action prediction error (APE) compared to the 256-token baseline, while providing 5.2× faster generation. APE measures how accurately an IDM (trained as in Tab. 3) can recover the conditioning action from generated video frames—a metric that directly evaluates whether the world model captures action-driven dynamics. The substantial improvement in APE validates our hypothesis that CompACT's modular tokens, which naturally attend to dynamic objects like end-effector and manipulation targets (Fig. 4), are inherently better suited for modeling action-conditioned state transitions. Qualitative results (Fig. 6) further support this: videos generated from CompACT latents maintain consistent action-driven end-effector movements, while the target tokenizer fails to preserve these dynamics.

Table 6. **Action-conditioned video prediction results on RoboNet [14].** Action prediction error is measured using IDM reported in Tab 3. Latency is measure for when generating next 14 frames on a single RTX 6000 ADA GPU.

Model	#tok	APE	Latency (sec)
Target tokenizer ($\mathcal{D}_\psi \circ \mathcal{E}_\psi$) [8]	256	0.3383	3.826
CompACT	16	0.1122	0.740

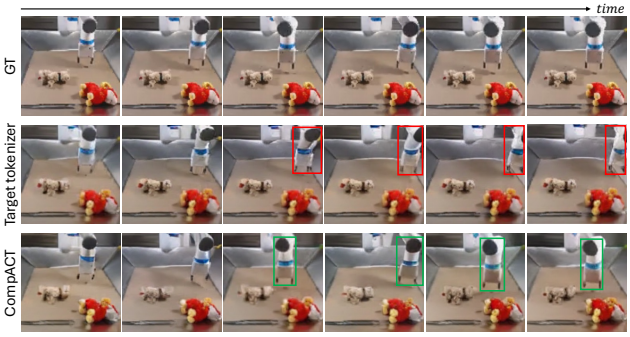

Figure 6. **Qualitative results of action-conditioned video generation.** Red and green boxes indicate incorrect and correct end-effector positions, respectively.

5. Conclusion

In this work, we present CompACT, a compact tokenizer that achieves extreme compression by representing images with only 16 or 8 discrete tokens while preserving planning-critical information. The key insight enabling this compression is our use of frozen vision foundation models as the encoder backbone: by leveraging pretrained semantic representations, our tokenizer naturally prioritizes high-level spatial and semantic features over reconstruction-oriented details. We demonstrate that world models trained in this compact latent space outperform baselines requiring larger token counts while achieving a 40× speedup in planning, validating our hypothesis that effective planning requires semantic abstraction rather than photorealistic reconstruction.

Planning in 8 Tokens: A Compact Discrete Tokenizer for Latent World Model

Supplementary Material

This supplementary material provides detailed implementation specifics and additional experimental results for CompACT. We organize the content as follows: Sec. 6 describes the complete CompACT tokenizer architecture, training procedure, and hyperparameters. Sec. 7 details the Inverse Dynamics Model (IDM) used for action prediction experiments on RoboNet [14]. Sec. 8 presents the world model architectures for both navigation (autoregressive with fixed history window) and manipulation tasks (block-causal parallel prediction). Sec. 9 explains the Cross-Entropy Method used for navigation planning. Sec. 10 provides additional qualitative results including reconstruction examples, attention visualizations, and planning rollouts. Sec. 11 provides comparison between tokenizers in terms of planning efficiency. Finally, Sec. 12 presents that with compact latent tokens we can scale up the world model while remain efficient.

Table 7. **Training hyperparameters for CompACT.**

hparams	CompACT
optimizer	AdamW
β_1	0.9
β_2	0.999
weight decay	0.01
lr	0.0001
lr scheduling	cosine
lr warmup steps	10K
batch size (ImageNet)	512
training steps (ImageNet)	500K
training steps (RoboNet finetuning [14])	256
training steps (RoboNet finetuning [14])	100K

Table 8. **Model architecture hyperparameters for CompACT.**

hparams	Latent resampler	$\mathcal{D}_{\text{compact}}$
depth	5	16
dim	768	1024
MLP dim	3072	4096
heads	8	8

6. Details of CompACT tokenizer

Tab. 7 and Tab. 8 summarize the training and model architecture hyperparameters of CompACT, respectively.

Tokenizer architecture. We use frozen DINOv3-B [59] in the encoder $\mathcal{E}_{\text{compact}}$. In DINOv3-B, we re-initialized the last layer normalization's affine parameters (weight and bias to 1 and 0, respectively). We found that using pretrained affine parameters as-is results in codebook collapse during training, since the output of $\mathcal{E}_{\text{compact}}$ has specific statistics when using pretrained layernorm affine parameters. For the latent resampler, we use the 5 transformer decoder blocks similar to DETR [7] and Perceiver [37]. Number of learnable queries determines the number of latent tokens used in CompACT. We use the Finite Scalar Quantization [48] (FSQ) for discretizing the output of latent resampler. Levels per channel are set to $[8, 8, 8, 5, 5, 5]$, which is the recommended level configuration to construct approximately 2^{16} size codebook. For decoder $\mathcal{D}_{\text{compact}}$, we use the MM-DiT [20], which takes compact latent as a condition. Fig. 7 depicts the details architecture of the decoder $\mathcal{D}_{\text{compact}}$, which is shown in simplified form in Fig. 2 in the main paper. $\mathcal{D}_{\text{compact}}$ is trained from the scratch. Proposed CompACT comprises 775M parameters in total, including $\mathcal{E}_{\text{compact}}$, $\mathcal{D}_{\text{compact}}$, and $\mathcal{D}_{\psi}$.

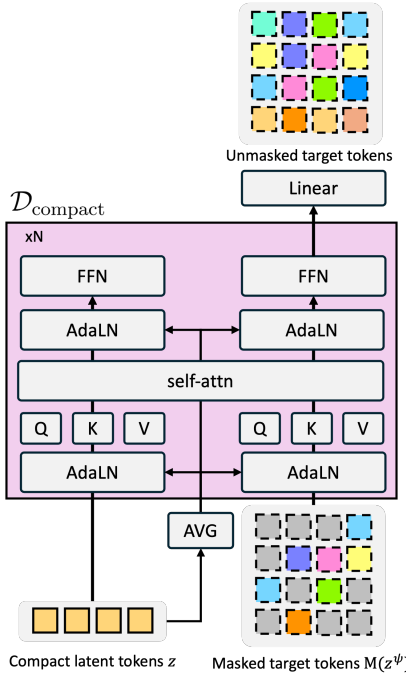

Figure 7. **$\mathcal{D}_{\text{compact}}$ architecture.** The proposed $\mathcal{D}_{\text{compact}}$ is based on MM-DiT [20], a DiT [54] variant designed for multimodal input processing. The architecture consists of two parallel processing streams: one for compact latent tokens $\boldsymbol{z}$ and another for target latent tokens $\boldsymbol{z}^{\psi}$, which are fused through self attention over the concatenated token sequence. For the overall depiction of CompACT, we refer to Fig. 2 in the main paper.

Masked generative modeling. As noted in Sec. 3.2.2 of the main paper, $\mathcal{D}_{\text{compact}}$ is formulated as a masked generative model. During training, we randomly sample the mask ratio from $(0, 1]$ following the cosine masking schedule [8].

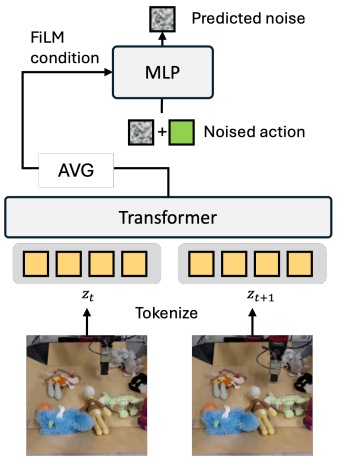

Figure 8. **Inverse Dynamics Model (IDM) architecture.** Consecutive frames are tokenized and processed through a transformer-based frame encoder, which produces a single conditioning vector via average pooling. This vector conditions an action denoiser implemented as a diffusion policy [12], which predicts the action taken between the two frames.

Table 9. **Model architecture hyperparameters for IDM.**

hparams	Frame encoder	hparams	Action denoiser
depth	4	# linear	4
dim	512	hidden dim	512
MLP dim	2048	diffusion	DDPM [35]
heads	8	timesteps	1000
#params	13.5M	#params	3.6M

During inference, we decode target latent tokens z^{ψ} by progressively unmasking them from a fully masked sequence, using compact latent tokens z as conditioning. In each iteration, a fraction of predictions with high confidence is accepted, while remaining tokens are re-masked. We follow the same cosine masking schedule during inference.

Dataset. CompACT tokenizer is trained on ImageNet-1K [16], on 224×224 (for fair comparison in navigation experiment following NWM [3]) and 256×256 resolution. For the augmentation, we used random crop and random horizontal flip. We additionally finetune the tokenizer for the experiment with RoboNet [14] since robot data has a significant domain gap compared to the ImageNet data. For RoboNet finetuning we used center cropped 256×256 resolution images.

Training and inference. For ImageNet [16] pretraining, CompACT is trained for 500K steps with batch size of 512. We use the AdamW [46] with $1e-4$ learning rate. β_1 and β_2 are set to 0.9 and 0.999, respectively. During inference, sampling step in $\mathcal{D}_{\text{compact}}$ is set to 16.

Table 10. **Model architecture hyperparameters for world model.**

hparams	navigation	manipulation
depth	12	16
dim	768	1024
MLP dim	3072	4096
heads	12	16
# params	243M	270M

7. Details of inverse dynamics model (IDM)

Fig. 8 and Tab. 9 present the model architecture and hyperparameters of IDM, respectively.

Architecture. The Inverse Dynamics Model (IDM) used for RoboNet experiments consists of two modules: a frame encoder and an action denoiser. Given latent tokens from two consecutive frames, the frame encoder first processes the concatenated token sequence through a 4-layer transformer encoder. The target tokenizer [8] requires processing 512 tokens, while CompACT requires only 32 tokens—a 16× reduction in sequence length. The frame encoder output is average-pooled into a single vector, which serves as the conditioning signal for the action denoiser. The action denoiser is a multi-layer perceptron (MLP) with 4 linear layers (hidden dimension 512), SiLU activation [33], and FiLM conditioning [55] between each layer. The denoiser takes a noisy 5-dimensional action and predicts the applied noise. Following diffusion policy [12], we use DDPM [35] with a squared cosine schedule [51].

Dataset. IDM is trained on RoboNet [14]. Frame pairs are randomly sampled at each iteration. Frames are pre-encoded using tokenizer (CompACT or target tokenizer) with 256×256 resolution center-cropped images. 250 episodes are used for test.

Training and inference. IDM is trained for 100K steps with batch size of 128. We use the AdamW [46] with $1e-4$ learning rate. β_1 and β_2 are set to 0.9 and 0.999, respectively. During inference, we use 1000 diffusion timesteps for sampling.

Evaluation. We evaluate IDM performance using two metrics computed on predicted end-effector positions. First, we measure L1 error between the predicted and ground truth end-effector positions. Second, we compute the coefficient of determination (R^2), which measures how well the model explains the variance in end-effector movements. R^2 ranges from 0 to 1, where higher values indicate better prediction. For instance, $R^2 = 0.9$ means that model can explain 90% of the variance in end-effector movements.

8. World model details

8.1. Navigation: autoregressive with fixed history window

For navigation tasks, we adopt an autoregressive formulation where the world model predicts the next latent state z_{t+1}

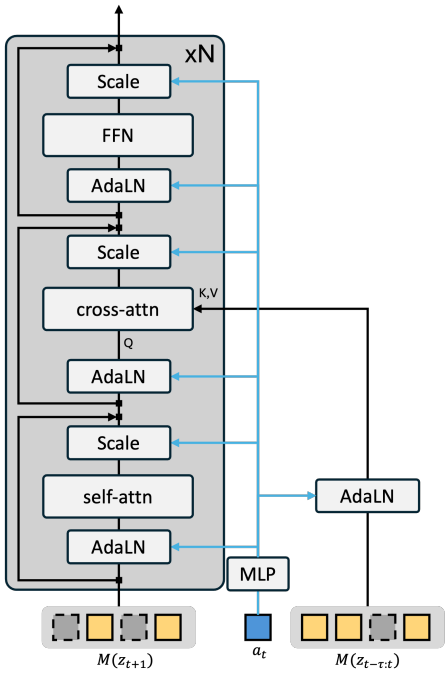

Figure 9. **World model f_ϕ architecture for navigation.** For navigation tasks, the world model is formulated as an action-conditioned generative model that autoregressively generates the next frame. The model follows the CDiT architecture proposed in NWM [3], where actions are conditioned via adaptive layer normalization and history frames are conditioned via cross-attention.

conditioned on a fixed-length history window of past states $\{z_{t-\tau}, \ldots, z_t\}$ and the action a_t. This design follows the Navigation World Models (NWM) [3] framework but operates in our compact discrete latent space with $N \leq 16$ tokens per timestep. Fig. 9 and Tab. 10 present the model architecture and hyperparameters of world model for navigation, respectively.

Architecture. Fig. 9 illustrates the architecture. We employ a DiT-based [54] architecture with multiple transformer blocks, where each block consists of adaptive layer normalization (AdaLN), multi-head self-attention, cross-attention, and feed-forward networks (FFN). The action a_t is first encoded through an MLP and used to modulate the transformer blocks via AdaLN, similar to how DiT conditions on class labels. The model takes as input the masked future tokens $M(z_{t+1})$ and attends to the history window $M(z_{t-\tau:t})$ through cross-attention layers, where $M(\cdot)$ denotes the masking operation. The history tokens serve as keys and values (K, V), while the future tokens act as queries (Q). We use $\tau = 4$ for the history window length and $N = 12$ for the DiT blocks. For SD-VAE experiments, we follow the exact architecture from NWM [3] (CDiT-B). For discrete tokenizers (FlexTok [2] and CompACT), we use the same configuration with two adaptations to handle discrete tokens: (1) embedding and classification layers instead of continuous projections, and (2) masked generative modeling instead of diffusion-based generation.

History masking. Following the principles of diffusion forcing [10], we randomly mask tokens in the history window during training. This encourages the model to learn robust temporal dependencies and improves action conditioning. During each training iteration, we randomly mask tokens in each historical frame. At inference time, we use the slightly masked (20%) history for stable rollouts. Ablation results in Table 5 of the main paper demonstrate that history masking improves planning accuracy.

Dataset. The navigation world model is trained on three datasets: RECON [58], SCAND [39], and HuRoN [34]. For HuRoN, we use the publicly available low-resolution version. All datasets contain first-person navigation trajectories with corresponding actions (changes in x-axis, y-axis, and yaw). Frames are center-cropped and resized to 224×224 resolution. We follow the same train/test splits as NWM [3]. We exclude Tartan [66] and Ego4D [27], which were used in the original NWM [3], for the following reasons: Tartan consists predominantly of forward-only actions, and Ego4D, while providing large-scale data, is computationally expensive to process. We found that we can fairly reproduce navigation planning performance without them.

Reproducing NWM [3]. The SD-VAE row in Table 4 of the main paper can be considered as our reproduction of NWM [3]. The original NWM reports ATE of 1.13 and RPE of 0.35 on RECON, while our reproduction achieves ATE of 1.262 and RPE of 0.354. Despite several differences—using a smaller model (CDiT-B instead of CDiT-XL used in the original NWM), low-resolution data for HuRoN, and excluded datasets (Tartan and Ego4D)—we fairly reproduce the navigation planning performance.

Training and inference. The navigation world model is trained for 200K steps with a batch size of 128. We use AdamW optimizer [46] with a learning rate of 1×10^{-4}. We set $\beta_1 = 0.9$ and $\beta_2 = 0.999$, with weight decay of 1×10^{-2}. For the masked generative modeling objective, we randomly sample the mask ratio from $(0, 1]$ following the cosine masking schedule [8]. During inference, we used 8 and 4 sampling steps for 16 and 8 latent tokens, respectively.

Evaluation. We evaluate navigation planning performance using two standard trajectory metrics: Absolute Trajectory Error (ATE) and Relative Pose Error (RPE) [60]. First, ATE measures the RMSE between predicted and ground truth transformations from the initial frame across the entire trajectory between corresponding timesteps. Second, RPE measures the error in relative transformations between consecutive timesteps. Lower values indicate better planning performance for both metrics. For details of model-predictive control framework used for navigation planning, we refer Sec. 9.

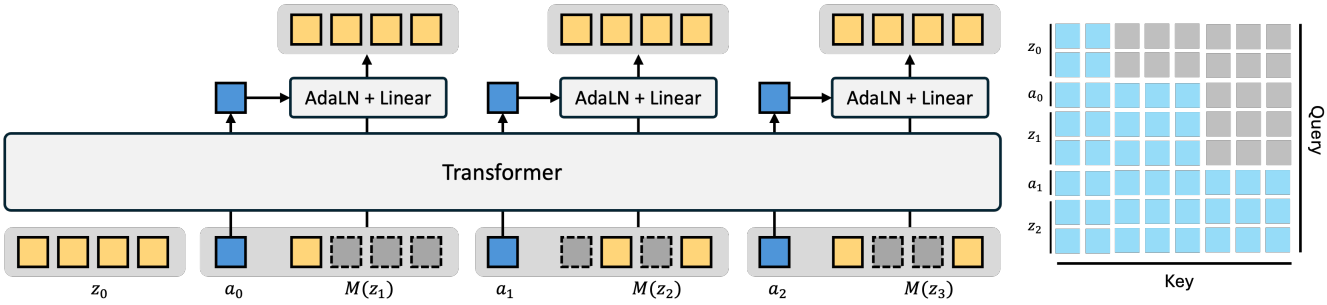

Figure 10. **World model f_ϕ architecture for manipulation.** For manipulation tasks, the world model is formulated as a block-causal transformer that enables parallel prediction of multiple future frames conditioned on historical timesteps. Action tokens are used to condition the prediction head (AdaLN + linear layer) that unmasks latent tokens at each future timestep.

8.2. Manipulation: block-causal parallel prediction

For robotic manipulation tasks on RoboNet [14], we adopt a block-causal transformer architecture that predicts multiple future frames $\{z_{t+1}, \ldots, z_{t+K}\}$ in parallel, conditioned on the initial observation z_t and action sequence $\{a_t, \ldots, a_{t+K-1}\}$. This parallel formulation is more suitable for video generation tasks where we need to produce extended action-conditioned rollouts efficiently. Fig. 10 and Tab. 10 present the model architecture and hyperparameters of world model for manipulation, respectively.

Architecture. Figure 10 illustrates the block-causal architecture. Unlike the autoregressive model, all future frames are processed simultaneously within a single transformer. The input sequence is constructed by interleaving observations and actions: $[z_{t-\tau:t}, a_t, M(z_{t+1}), \ldots, a_{t+H-1}, M(z_{t+H})]$, where $M(\cdot)$ denotes masked tokens. The key component is the block-causal attention mask (shown on the right of Figure 10): tokens at timestep $t+i$ can attend to all tokens up to and including z_{t+i} and a_{t+i-1}, but cannot attend to future observations or actions. This preserves causal structure while enabling parallel prediction. Each action is encoded through a linear layer. The output token corresponding to each action is then used to condition a prediction head (AdaLN + Linear layer), which produces the unmasked tokens for the subsequent frame. During training, we set the prediction horizon to $H = 14$ (predicting the next 14 frames in parallel). However, the model can generalize to arbitrary horizon lengths since we use causal masking. We provide the first two frames of each episode as context for the model to condition on ($\tau = 2$).

Baseline configuration. For fair comparison, experiments using the target tokenizer (MaskGIT-VQGAN [8]) employ the same block-causal architecture. The only difference is the sequence length: the target tokenizer processes 256 tokens per frame, while CompACT processes only 16 tokens per frame, resulting in significantly faster generation (Tab. 6 in the main paper).

Diffusion forcing interpretation. This architecture naturally implements causal, discrete extension of diffusion forcing [10]: during training, masked tokens at different timesteps provide varying levels of noisy conditioning to future frames. A frame with fewer masked tokens acts as a cleaner conditioning signal, while heavily masked frames force the model to rely more on learned dynamics and action conditioning. This training scheme improves the model's ability to generate consistent action-conditioned video sequences.

Dataset. The manipulation world model is trained on RoboNet [14], similar to IDM experiment. Frames are pre-encoded using robonet finetuned CompACT with 256 $\times$ 256 resolution center-cropped images. For the target tokenizer [8] baseline, we use it as-is without RoboNet finetuning, which is fair since finetuned CompACT also keeps the frozen target tokenizer. 250 episodes are used for test.

Training and inference. The manipulation world model is trained for 100K steps with a batch size of 128. We use AdamW optimizer [46] with a learning rate of 1×10^{-4}. We set $\beta_1 = 0.9$ and $\beta_2 = 0.999$, with weight decay of 1×10^{-2}. For the masked generative modeling objective, we randomly sample the mask ratio from $(0, 1]$ following the cosine masking schedule [8]. We used 100 sampling steps to generate future 14 frames.

9. Planning with cross-entropy method

As described in Sec. 3.1 and Sec. 3.3, we use a trained world model to standalone-plan goal-conditioned navigation trajectories by optimizing distance between final prediction and the goal. Here, we provide additional details about the optimization using the Cross-Entropy Method (CEM) [13, 15] and the hyperparameters used.

Cross-Entropy Method. For CEM planning, we strictly follow the protocol of NWM [3] for fair comparison. Algo. 1 presents the complete CEM procedure. Specifically, trajectory is assumed to be a straight line and only its endpoint is optimized, represented by three variables: a single translation u and yaw rotation ϕ. This is converted into an action

Algorithm 1 Cross-Entropy Method for Navigation Planning

Require: Initial frame o_0, goal frame o_{goal}, planning horizon H, World model f_ϕ, tokenizer $\mathcal{D}_{\text{compact}} \circ \mathcal{E}_{\text{compact}}$
Require: CEM params: population size N, top samples to be selected K, iterations I, samples per candidate M

// Initialize action distribution
$\mu \leftarrow (\mu_x, \mu_y, \mu_\phi)$
$\Sigma \leftarrow \text{diag}(\sigma_x^2, \sigma_y^2, \sigma_\phi^2)$
Encode observations: $z_0 \leftarrow \mathcal{E}_{\text{compact}}(o_0)$, $z_{\text{goal}} \leftarrow \mathcal{E}_{\text{compact}}(o_{\text{goal}})$
for $i = 1$ to I **do**
  $\mathcal{A} \leftarrow \{(u_j, \phi_j)\}_{j=1}^{N}$ where $(u_j, \phi_j) \sim \mathcal{N}(\mu, \Sigma)$ // Sample candidate actions
  for each candidate $(u_j, \phi_j) \in \mathcal{A}$ **do**
    // Evaluate via stochastic rollouts
    for $m = 1$ to M **do**
      $z_t^{(m)} \leftarrow z_0$
      $\mathbf{a}^{(j)} \leftarrow \texttt{ToActionSeq}(u_j, \phi_j, H)$
      for $t = 0$ to $H - 1$ **do**
        $z_{t+1}^{(m)} \sim f_\phi(z_t^{(m)}, a_t^{(j)})$ // World model rollout
      end for
      $\hat{o}_H^{(m)} \leftarrow \mathcal{D}_\psi \circ \mathcal{D}_{\text{compact}}(z_H^{(m)})$
      $\hat{o}_{\text{goal}} \leftarrow \mathcal{D}_\psi \circ \mathcal{D}_{\text{compact}}(z_{\text{goal}})$
      $c_m \leftarrow d(\hat{o}_H^{(m)}, \hat{o}_{\text{goal}})$ or $c_m \leftarrow d(z_H^{(m)}, z_{\text{goal}})$// LPIPS between reconstruction or L1 distance between latent
    end for
    $C_j \leftarrow \frac{1}{M} \sum_{m=1}^{M} c_m$
  end for
  $\tilde{\mathcal{A}} \leftarrow$ top-K from $\mathcal{A}$ with lowest costs $\{C_j\}$ // Select top-K candidates
  // Update distribution parameters
  $\mu \leftarrow \frac{1}{K} \sum_{(u,\phi) \in \tilde{\mathcal{A}}} (u, \phi)$
  $\Sigma \leftarrow \frac{1}{K} \sum_{(u,\phi) \in \tilde{\mathcal{A}}} ((u, \phi) - \mu)((u, \phi) - \mu)^\top$
end for
return Action with minimum cost from $\tilde{\mathcal{A}}$

sequence by evenly distributing the translation across H timesteps and applying the rotation only at the final step. Each action step corresponds to a fixed time interval of 0.25 seconds.

Hyperparameters. We set the population size to 80 since increasing over it does not lead to substantial gain in planning accuracy. For other hyperparameters, we follow the configuration of NWM: 2 seconds planning ($H = 8$), single iteration ($I = 1$), top-K selection with $K = 5$, and $M = 3$ repetitive stochastic rollouts per candidate action.

Distance metric $d(\cdot, \cdot)$. The distance between predicted and goal observations can be measured in either pixel space or latent space. For pixel-space evaluation, we decode latent tokens to images using $\mathcal{D}_\psi \circ \mathcal{D}_{\text{compact}}$ and compute the LPIPS distance [38]. Alternatively, we can compute distances directly in the discrete latent space, which offers significant computational speedup by avoiding the decoding step (Tab. 5 (*middle*) in the main paper). This latent-space distance computation is enabled by the properties of Finite Scalar Quantization (FSQ) [48], which we use for discretization. Unlike traditional vector quantization that uses arbitrary codebook indices, FSQ assigns each latent vector to a discrete code based on level-based radix representation, preserving the continuous geometric structure in the discrete space. Specifically, each dimension of the latent is quantized to one of L equally-spaced levels, allowing us to map discrete token indices back to their level-based radix representation. Given two discrete latent tokens $z_i, z_j \in \{1, \ldots, K\}$, we compute their distance as:

$$d(z_i, z_j) = \|\texttt{FSQ}^{-1}(z_i) - \texttt{FSQ}^{-1}(z_j)\|_1, \tag{6}$$

where $\texttt{FSQ}^{-1}(\cdot)$ maps the discrete index to its corresponding level-based radix representation. This distance in the discrete latent space provides a meaningful measure of semantic similarity while enabling faster planning with marginal degradation compared to pixel-space metrics.

10. More qualitative results

Fig. 11 shows reconstruction examples across ImageNet, RECON, and RoboNet. While CompACT discards fine-grained textures and lighting details due to extreme compression, it preserves semantic content and spatial structure essential for planning tasks. Fig. 12 visualizes attention patterns in

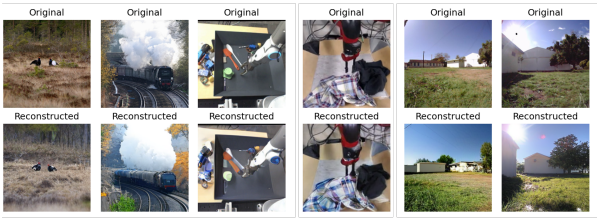

Figure 11. **Qualitative results of reconstruction with CompACT.**

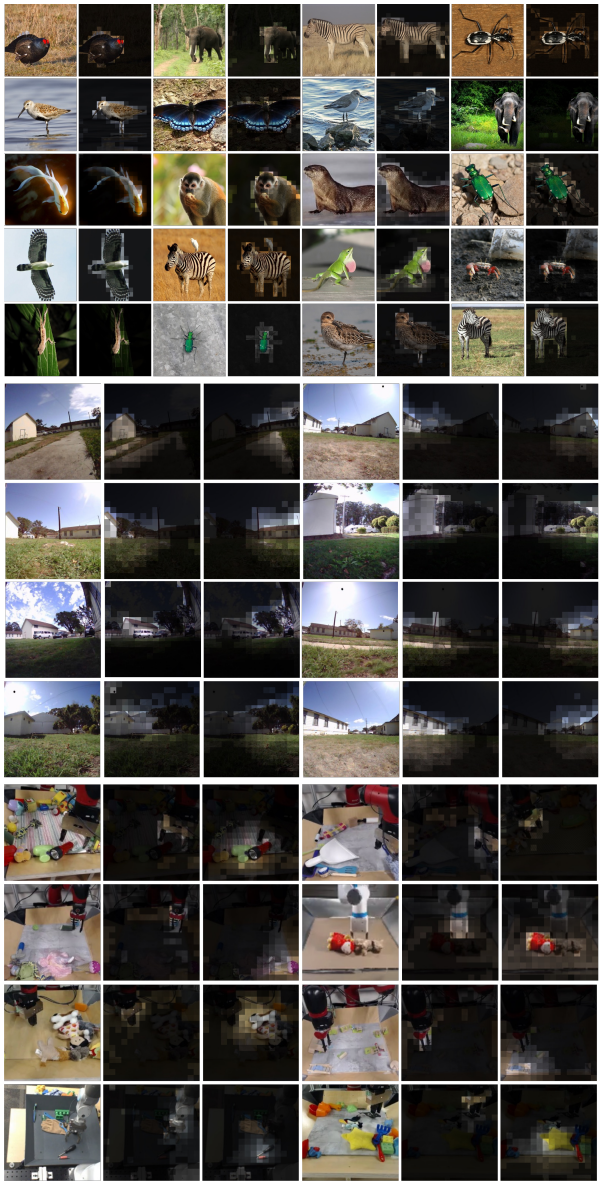

Figure 12. **Attention visualization for compact latent tokens in latent resampler.** Brighter the color, higher the attention score.

the latent resampler, demonstrating that each compact token attends to semantically coherent regions. Fig. 14 presents example planning results with CompACT. While fine-grained details such as textures and shadows are synthesized rather than reconstructed, the rollouts accurately preserve planning-critical information: spatial layout, object positions, and scene structure necessary for effective goal-reaching. Fig. 15 presents additional examples of action-conditioned video generation on RoboNet. Videos generated from CompACT latents maintain more consistent action-driven end-effector movements throughout the rollout compared to the ones generated with target tokenizer latents, validating that the modular latent tokens effectively capture dynamics-relevant information for manipulation tasks.

11. Planning efficiency analysis

Fig. 13 visualizes the trade-off between planning accuracy (ATE), planning latency, and model size across different tokenizers on RECON. Bubble size represents the peak VRAM usage during planning. CompACT variants achieve superior efficiency: CompACT delivers up to 80× speedup over SD-VAE while maintaining comparable accuracy. In contrast, FlexTok variants suffer significant accuracy loss and large VRAM requirements despite similar token counts, validating that our tokenizer design is critical for enabling efficient real-time planning.

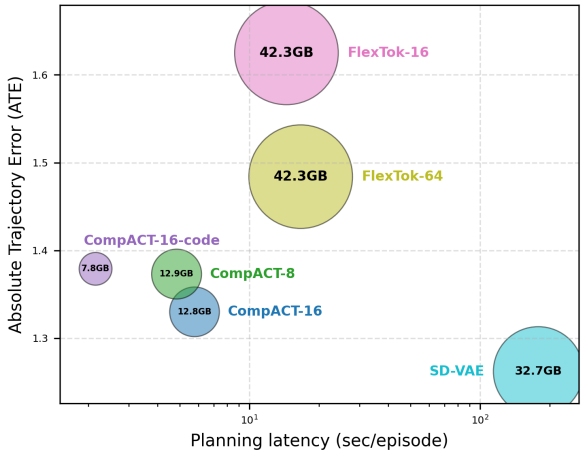

Figure 13. **Plot for ATE, planning latency, and memory peak usage on RECON [58].** Latency and memory usage is mesaured for single trajectory optimization, using a single RTX 6000 ADA GPU.

12. Scaling up with fewer tokens

The compact latent representation of CompACT enables scaling world models to larger capacities while maintaining practical planning latency. We train a 750M-parameter variant of our world model for navigation by increasing the depth to 24 layers and hidden dimension to 1024. This scaled model achieves improved planning accuracy with ATE of 1.305 and RPE of 0.370 on RECON—outperforming our base 16-token model (ATE=1.330, RPE=0.390). Planning latency is 24.7 seconds per trajectory, still 7× faster than the SD-VAE baseline (178.78 seconds).

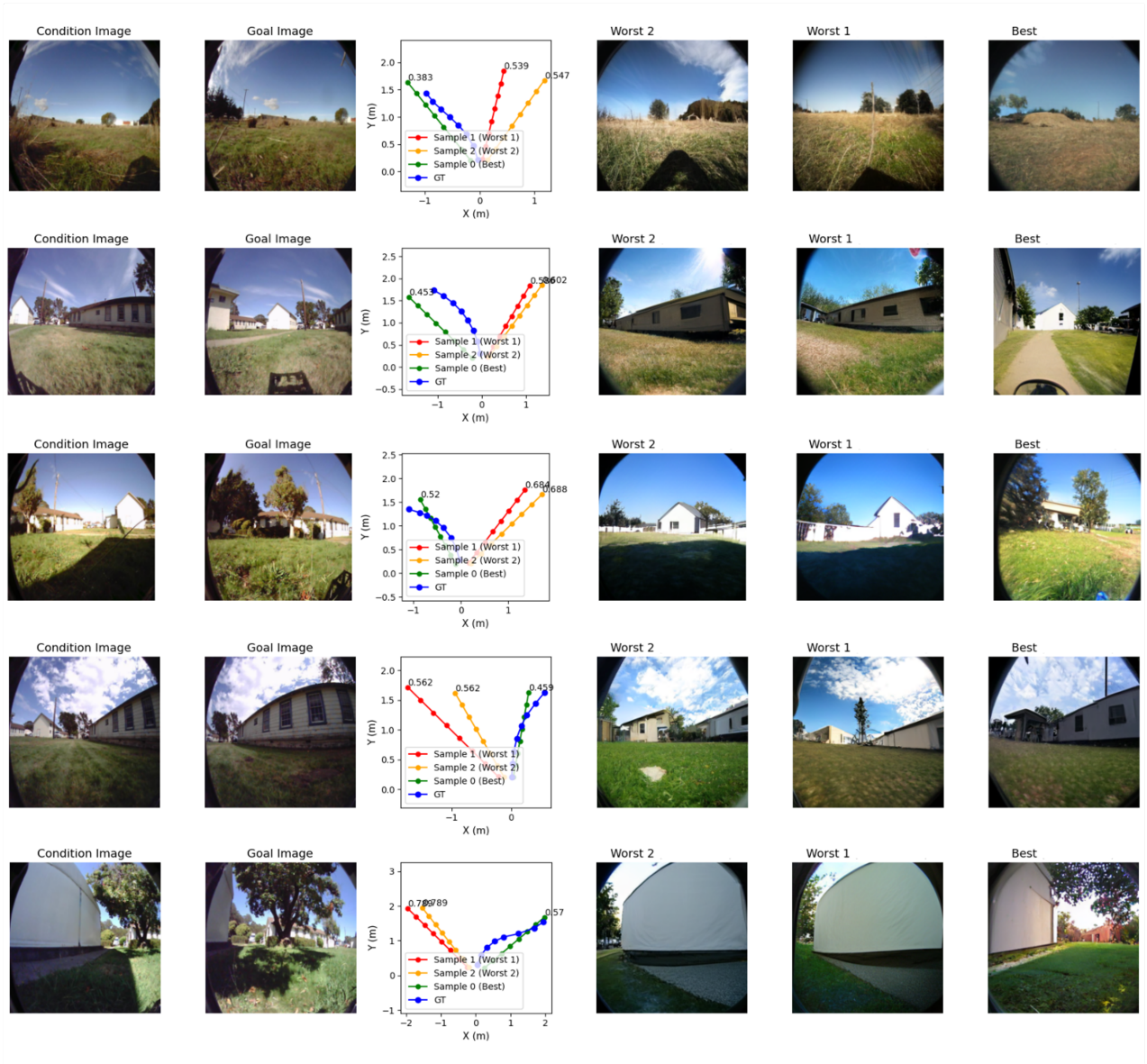

Figure 14. **Additional qualitative results of navigation planning with the proposed CompACT.** Among candidates, worst two action sequences and best action sequences are presented together.

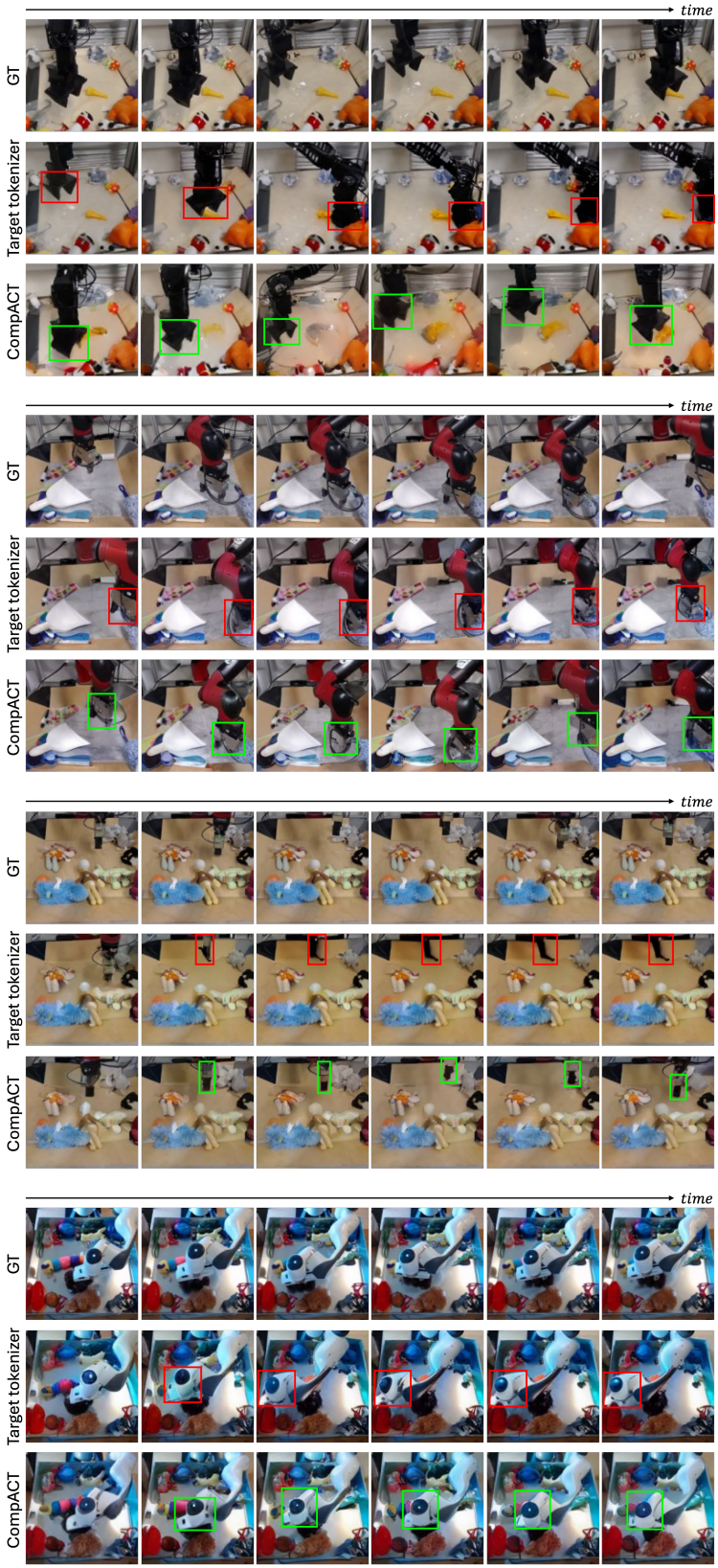

Figure 15. **Additional qualitative results of action-conditioned video generation.** Red and green boxes indicate incorrect and correct end-effector positions, respectively.

References

[1] 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. 1, 3

[2] Roman Bachmann, Jesse Allardice, David Mizrahi, Enrico Fini, Oğuzhan Fatih Kar, Elmira Amirloo, Alaaeldin El-Nouby, Amir Zamir, and Afshin Dehghan. Flextok: Resampling images into 1d token sequences of flexible length. In *Forty-second International Conference on Machine Learning*, 2025. 2, 6, 7, 3

[3] Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 15791–15801, 2025. 1, 3, 4, 5, 6, 2

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *NeurIPS*, 2020. 5

[5] Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. In *Forty-first International Conference on Machine Learning*, 2024. 3

[6] Shiyue Cao, Yueqin Yin, Lianghua Huang, Yu Liu, Xin Zhao, Deli Zhao, and Kaigi Huang. Efficient-vqgan: Towards high-resolution image generation with efficient vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7368–7377, 2023. 2

[7] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229. Springer, 2020. 1

[8] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *CVPR*, 2022. 1, 2, 3, 5, 6, 7, 8, 4

[9] Huiwen Chang, Han Zhang, Jarred Barber, AJ Maschinot, José Lezama, Lu Jiang, Ming-Hsuan Yang, Kevin Murphy, William T Freeman, Michael Rubinstein, et al. Muse: Text-to-image generation via masked generative transformers. In *ICML*, 2023. 3

[10] 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. 5, 3, 4

[11] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *ICML*, 2020. 3

[12] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023. 2

[13] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems*, 31, 2018. 4

[14] Sudeep Dasari, Frederik Ebert, Stephen Tian, Suraj Nair, Bernadette Bucher, Karl Schmeckpeper, Siddharth Singh, Sergey Levine, and Chelsea Finn. Robonet: Large-scale multi-robot learning. *arXiv preprint arXiv:1910.11215*, 2019. 2, 5, 6, 7, 8, 1, 4

[15] Pieter-Tjerk De Boer, Dirk P Kroese, Shie Mannor, and Reuven Y Rubinstein. A tutorial on the cross-entropy method. *Annals of operations research*, 134(1):19–67, 2005. 1, 3, 4

[16] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 6, 2

[17] Yilun Du, Mengjiao Yang, Pete Florence, Fei Xia, Ayzaan Wahid, Brian Ichter, Pierre Sermanet, Tianhe Yu, Pieter Abbeel, Joshua B Tenenbaum, et al. Video language planning. *arXiv preprint arXiv:2310.10625*, 2023. 1

[18] Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. *Advances in neural information processing systems*, 36:9156–9172, 2023. 3

[19] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12873–12883, 2021. 1, 2

[20] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *ICML*, 2024. 1

[21] Cunxin Fan, Xiaosong Jia, Yihang Sun, Yixiao Wang, Jianglan Wei, Ziyang Gong, Xiangyu Zhao, Masayoshi Tomizuka, Xue Yang, Junchi Yan, et al. Interleave-vla: Enhancing robot manipulation with interleaved image-text instructions. *arXiv preprint arXiv:2505.02152*, 2025. 3

[22] Lijie Fan, Tianhong Li, Siyang Qin, Yuanzhen Li, Chen Sun, Michael Rubinstein, Deqing Sun, Kaiming He, and Yonglong Tian. Fluid: Scaling autoregressive text-to-image generative models with continuous tokens. *arXiv preprint arXiv:2410.13863*, 2024. 3

[23] Jay W Forrester. Counterintuitive behavior of social systems. *Theory and decision*, 2(2):109–140, 1971. 1

[24] Shanghua Gao, Pan Zhou, Ming-Ming Cheng, and Shuicheng Yan. Mdtv2: Masked diffusion transformer is a strong image synthesizer. *arXiv preprint arXiv:2303.14389*, 2023. 3

[25] Shenyuan Gao, Jiazhi Yang, Li Chen, Kashyap Chitta, Yihang Qiu, Andreas Geiger, Jun Zhang, and Hongyang Li. Vista: A generalizable driving world model with high fidelity and versatile controllability. *Advances in Neural Information Processing Systems*, 37:91560–91596, 2024. 3

[26] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 2

[27] 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 *CVPR*, pages 18995–19012, 2022. 3

[28] David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2(3), 2018. 1, 3

[29] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019. 1

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

[31] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023. 1

[32] Nicklas Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control. *arXiv preprint arXiv:2310.16828*, 2023. 1, 3

[33] D Hendrycks. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 2

[34] 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. 6, 3

[35] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, pages 6840–6851, 2020. 2

[36] Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. Gaia-1: A generative world model for autonomous driving. *arXiv preprint arXiv:2309.17080*, 2023. 3

[37] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *International conference on machine learning*, pages 4651–4664. PMLR, 2021. 1

[38] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016. 4, 5

[39] 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. 6, 3

[40] Dongwon Kim, Ju He, Qihang Yu, Chenglin Yang, Xiaohui Shen, Suha Kwak, and Liang-Chieh Chen. Democratizing text-to-image masked generative models with compact text-aware one-dimensional tokens. 2025. 2, 6

[41] Po-Chen Ko, Jiayuan Mao, Yilun Du, Shao-Hua Sun, and Joshua B Tenenbaum. Learning to act from actionless videos through dense correspondences. *arXiv preprint arXiv:2310.08576*, 2023. 3

[42] 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. 3

[43] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11523–11532, 2022. 2

[44] Tianhong Li, Huiwen Chang, Shlok Mishra, Han Zhang, Dina Katabi, and Dilip Krishnan. Mage: Masked generative encoder to unify representation learning and image synthesis. In *CVPR*, 2023. 3

[45] Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. *NeurIPS*, 2024. 3

[46] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 2, 3, 4

[47] Russell Mendonca, Shikhar Bahl, and Deepak Pathak. Structured world models from human videos. *arXiv preprint arXiv:2308.10901*, 2023. 3

[48] Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. Finite scalar quantization: Vq-vae made simple. *arXiv preprint arXiv:2309.15505*, 2023. 2, 4, 1, 5

[49] Vincent Micheli, Eloi Alonso, and François Fleuret. Transformers are sample-efficient world models. *arXiv preprint arXiv:2209.00588*, 2022. 1

[50] Keita Miwa, Kento Sasaki, Hidehisa Arai, Tsubasa Takahashi, and Yu Yamaguchi. One-d-piece: Image tokenizer meets quality-controllable compression. *arXiv preprint arXiv:2501.10064*, 2025. 2

[51] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021. 2

[52] Dujun Nie, Xianda Guo, Yiqun Duan, Ruijun Zhang, and Long Chen. Wmnav: Integrating vision-language models into world models for object goal navigation. *arXiv preprint arXiv:2503.02247*, 2025. 3

[53] Gaurav Parmar, Richard Zhang, and Jun-Yan Zhu. On aliased resizing and surprising subtleties in gan evaluation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11410–11420, 2022. 6

[54] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, 2023. 1, 4, 5, 3

[55] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, 2018. 2

[56] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 1, 2, 6, 7

[57] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *NeurIPS*, 2016. 6

[58] Dhruv Shah, Benjamin Eysenbach, Nicholas Rhinehart, and Sergey Levine. Rapid exploration for open-world navigation with latent goal models. In *5th Annual Conference on Robot Learning*, 2021. 2, 6, 3

[59] Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025. 2, 4, 6, 1

[60] Jürgen Sturm, Wolfram Burgard, and Daniel Cremers. Evaluating egomotion and structure-from-motion approaches using the tum rgb-d benchmark. In *Proc. of the Workshop on Color-Depth Camera Fusion in Robotics at the IEEE/RJS International Conference on Intelligent Robot Systems (IROS)*, page 6, 2012. 3

[61] Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive model beats diffusion: Llama for scalable image generation. *arXiv preprint arXiv:2406.06525*, 2024. 5

[62] Haotian Tang, Yecheng Wu, Shang Yang, Enze Xie, Junsong Chen, Junyu Chen, Zhuoyang Zhang, Han Cai, Yao Lu, and Song Han. Hart: Efficient visual generation with hybrid autoregressive transformer. *arXiv preprint arXiv:2410.10812*, 2024. 3

[63] Dani Valevski, Yaniv Leviathan, Moab Arar, and Shlomi Fruchter. Diffusion models are real-time game engines. *arXiv preprint arXiv:2408.14837*, 2024. 3

[64] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. 2

[65] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 2017. 4

[66] Wenshan Wang, Delong Zhu, Xiangwei Wang, Yaoyu Hu, Yuheng Qiu, Chen Wang, Yafei Hu, Ashish Kapoor, and Sebastian Scherer. Tartanair: A dataset to push the limits of visual slam. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4909–4916. IEEE, 2020. 3

[67] Mark Weber, Lijun Yu, Qihang Yu, Xueqing Deng, Xiaohui Shen, Daniel Cremers, and Liang-Chieh Chen. Maskbit: Embedding-free image generation via bit tokens. *arXiv preprint arXiv:2409.16211*, 2024. 3

[68] Grady Williams, Paul Drews, Brian Goldfain, James M Rehg, and Evangelos A Theodorou. Aggressive driving with model predictive path integral control. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 1433–1440. IEEE, 2016. 1, 4

[69] Mengjiao Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. *arXiv preprint arXiv:2310.06114*, 1(2):6, 2023. 1, 3

[70] 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. 3

[71] Jiahui Yu, Xin Li, Jing Yu Koh, Han Zhang, Ruoming Pang, James Qin, Alexander Ku, Yuanzhong Xu, Jason Baldridge, and Yonghui Wu. Vector-quantized image modeling with improved vqgan. *arXiv preprint arXiv:2110.04627*, 2021. 6

[72] Lijun Yu, José Lezama, Nitesh B Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Vighnesh Birodkar, Agrim Gupta, Xiuye Gu, et al. Language model beats diffusion–tokenizer is key to visual generation. *arXiv preprint arXiv:2310.05737*, 2023. 2, 3, 5

[73] Qihang Yu, Mark Weber, Xueqing Deng, Xiaohui Shen, Daniel Cremers, and Liang-Chieh Chen. An image is worth 32 tokens for reconstruction and generation. *Advances in Neural Information Processing Systems*, 37:128940–128966, 2024. 2, 6

[74] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 2

[75] 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 *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 10412–10420, 2025. 3

[76] Chuanxia Zheng, Tung-Long Vuong, Jianfei Cai, and Dinh Phung. Movq: Modulating quantized vectors for high-fidelity image generation. *Advances in Neural Information Processing Systems*, 35:23412–23425, 2022. 3

[77] Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. Dino-wm: World models on pre-trained visual features enable zero-shot planning. *arXiv preprint arXiv:2411.04983*, 2024. 1

[78] Fangqi Zhu, Hongtao Wu, Song Guo, Yuxiao Liu, Chilam Cheang, and Tao Kong. Irasim: Learning interactive real-robot action simulators. *arXiv preprint arXiv:2406.14540*, 2024. 1