MinD: Unified Visual Imagination and Control via Hierarchical World Model

Xiaowei Chi[1,3*†], Kuangzhi Ge[2*], Jiaming Liu[2†], Siyuan Zhou[1,3], Peidong Jia[2], Zichen He[2],

Sirui Han[3], Yuzhen Liu[1], Tingguang Li[1], Lei Han[1], Shanghang Zhang[2 ✉], Yike Guo[3 ✉]

[1]Tencent Robotics X; [2] Peking University; [3] Hong Kong University of Science and Technology

* Equal contribution, † Project lead, ✉Corresponding author

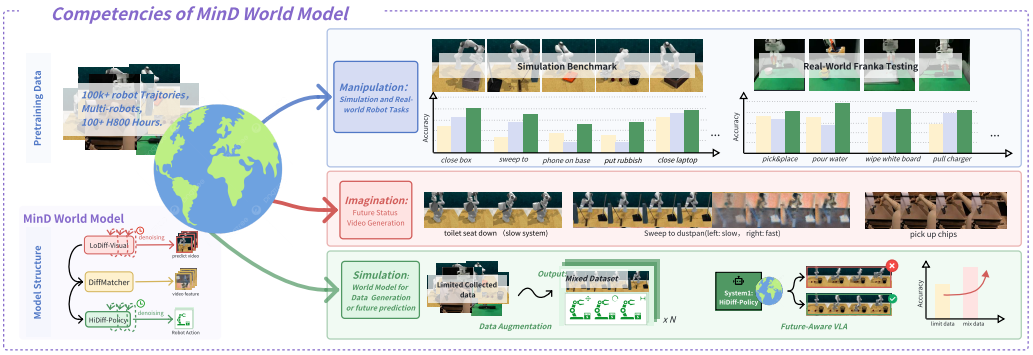

Figure 1: ***M***anipulate ***in D***ream(***MinD***): A video-action unified generation world model that can manipulate, imagine, and simulate. MinD integrates **LoDiff-Visual** for low-frequency video generation and **HiDiff-Policy** for high-frequency action planning. A dynamic feature adapter, **DiffMatcher**, bridges motion features between the two systems, ensuring consistency across video and action.

Abstract

Video generation models (VGMs) offer a promising pathway for unified world modeling in robotics by integrating simulation, prediction, and manipulation. However, their practical application remains limited due to (1) slow generation speed, which limits real-time interaction, and (2) poor consistency between imagined videos and executable actions. To address these challenges, we propose **Manipulate in Dream (MinD)**, a hierarchical diffusion-based world model framework that employs a dual-system design for vision-language manipulation. MinD executes VGM at low frequencies to extract video prediction features, while leveraging a high-frequency diffusion policy for real-time interaction. This architecture enables low-latency, closed-loop control in manipulation with coherent visual guidance. To better coordinate the two systems, we introduce a video-action diffusion matching module(DiffMatcher), with a novel co-training strategy that uses separate schedulers for each diffusion model. Specifically, we introduce a diffusion-forcing mechanism to DiffMatcher that aligns their intermediate representations during training, helping the fast action model better understand video-based predictions. Beyond manipulation, MinD also functions as a world simulator, reliably predicting task success or failure in latent space before execution. Trustworthy analysis further shows that VGMs can preemptively evaluate task feasibility and mitigate risks. Extensive experiments across multiple benchmarks demonstrate that MinD achieves state-of-the-art manipulation 63+% in RL-Bench, advancing the frontier of unified world modeling in robotics. Our demo page: https://manipulate-in-dream.github.io/

Preprint. Under review.

1 Introduction

World models aim to endow agents with the ability to simulate, predict, and plan based on internal representations of their environment [12]. In robotics, such models are critical for enabling intelligent behavior under uncertainty and for reasoning about long-horizon consequences of actions. Recent progress in generative modeling, especially video generation models (VGMs) [25, 1], has opened up new opportunities to build unified world models that directly incorporate perception, imagination, and control from raw visual inputs.

In parallel, the field of embodied AI has increasingly emphasized the need for unified models that can support perception, prediction, and decision-making in a coherent framework. By enabling agents to "imagine" the visual consequences of future actions, video generation models (VGMs) offer a promising route to this goal [1]. Recent work has demonstrated how VGMs can be conditioned on action sequences and language instructions, forming the basis for integrated systems that combine simulation, planning, and control [35, 38]. In this view, VGMs are not merely generative tools, but potential foundations for general-purpose world models.

Despite their potential, the existing VGMs world model faces two critical challenges that hinder their application in embodied tasks. First, their generation speed is often too slow for real-time decision-making, particularly due to the multi-step nature of diffusion-based video generation [6, 34]. Second, there is a lack of consistency between imagined visual trajectories and the action instructions needed to execute them, leading to a disconnect between planning and control [10, 4].

To address the limitations of existing VGMs in robotic manipulation, we propose Manipulate in Dream (MinD), a hierarchical diffusion-based world model for vision-language manipulation tasks. As shown in Fig. 1, MinD consists of two modules operating at different temporal resolutions: a low-frequency video generator (LoDiff-Visual) that imagines coherent visual futures, and a high-frequency diffusion policy (HiDiff-Policy) for real-time control. To connect these asynchronous processes, we introduce the Action-Video Diffusion Matching Model (DiffMatcher), which injects intermediate features from LoDiff into HiDiff as dynamic guidance. This design enables MinD to generate consistent visual predictions while supporting low-latency policy execution.

However, aligning LoDiff and HiDiff is challenging due to their separate noise schedules and temporal resolutions. When LoDiff and HiDiff are trained independently, the two models tend to diverge—each generating outputs in isolation, leading to semantic inconsistency between imagined videos and executed actions. This issue is further exacerbated during asynchronous inference, where HiDiff consumes LoDiff features from varying noise time steps. To improve alignment, we introduce a co-training strategy: inspired by the diffusion-forcing mechanism [5], we construct paired video features at different noise levels, and apply a spatio-temporal contrastive loss to supervise the DiffMatcher in extracting stable and semantically meaningful features across diffusion steps. Co-training helps the fast action model better understand video features on different time steps, and gives consistent and coherent planning.

We validate MinD across multiple manipulation benchmarks and demonstrate its superior performance in planning speed, visual-action consistency, and data efficiency. Our key contributions are:

(1) We propose MinD, a hierarchical diffusion-based world model that unifies visual imagination and action policy learning for robotic manipulation. MinD consists of a low-frequency video generator and a high-frequency action policy, enabling coherent visual prediction and real-time control. **(2)** We introduce a DiffMatcher module comprising two key components: the dual alignment mechanism for bridging asynchronous diffusion schedulers, and the diffusion-forcing technique for aligning latent representations across visual and policy streams via spatio-temporal contrastive learning. **(3)** We demonstrate that our method achieves state-of-the-art performance across multiple manipulation benchmarks. Furthermore, through cross-analysis, we show that video observations can serve as reliable predictors of task success or failure.

2 Related Work

World Models for Embodied AI World models aim to provide agents with internal simulations of the environment to support prediction, planning, and decision-making [12]. Early efforts in this direction employed latent dynamics models for model-based reinforcement learning [33], while

more recent approaches integrate vision and language to support open-world reasoning [9, 3]. In the context of embodied AI, such models are increasingly required to generate coherent future states from raw visual observations, enabling long-horizon planning and closed-loop control. However, most prior work either focuses on purely reactive policies or lacks the ability to generate semantically meaningful future observations.

Video Generation Models for Imagination and Planning Video generation models (VGMs), especially those based on latent diffusion [24, 25], have recently shown strong potential in synthesizing realistic and temporally coherent video sequences [35, 6, 34, 8, 21]. These models can "imagine" multiple future possibilities conditioned on actions or language, making them attractive candidates for visual world modeling [38, 1, 16, 20, 37]. However, VGMs are often computationally expensive and suffer from long inference times, which limits their applicability in real-time robotics. Furthermore, their integration with downstream control policies remains underexplored, often resulting in disjointed simulation-control pipelines.

Multimodal Policy Learning and Vision-Language Manipulation Recent advancements in vision-language action (VLA) modeling have enabled robots to follow natural language instructions to perform manipulation tasks [22, 2]. Approaches such as diffusion policies [7] and transformer-based policies [18, 31] have shown promise in learning complex visuomotor behaviors. While these methods excel at conditioned action generation, they typically lack a forward model of the environment, limiting their planning capabilities. Some works attempt to bridge this gap by jointly modeling actions and visual outcomes [35, 20, 8, 15, 4, 32], but still struggle with temporal inconsistencies and lack of alignment between imagined futures and executable behaviors.

3 Preliminary: Diffusion Models

MinD is built on two diffusion models [13], which are used for both video generation [34] and policy learning [7]. We briefly describe the two variants used in our system.

3.1 Diffusion Models for Video and Policy

Our method employs two diffusion models with a shared denoising formulation: one for generating video sequences LoDiff-Visual, and the other for generating action sequences HiDiff-Policy. Both models learn to reverse a fixed forward noise process using a denoising objective.

Forward process. Given a clean target $\mathbf{x}_0$ (either video latent or action), the forward process adds Gaussian noise over T steps:

$$q(\mathbf{x}_\tau|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_\tau;\ \sqrt{\bar{\alpha}_\tau}\mathbf{x}_0,\ (1-\bar{\alpha}_\tau)\mathbf{I}), \tag{1}$$

where $\bar{\alpha}_\tau = \prod_{s=1}^{\tau}(1-\beta_s)$ and $\{\beta_\tau\}$ is a fixed variance schedule.

Reverse process. A neural network ϵ_θ is trained to predict the added noise from the noisy input $\mathbf{x}_\tau$, denoising timestep τ, and conditioning inputs:

$$\epsilon_\theta(\mathbf{x}_\tau, \tau, \mathbf{c}) \approx \epsilon, \tag{2}$$

The training objective minimizes the denoising error:

$$\mathcal{L}_{\text{diff}} = \mathbb{E}_{\mathbf{x}_0,\tau,\epsilon}\left[\left\|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_\tau}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_\tau}\epsilon,\ \tau,\ \mathbf{c})\right\|^2\right] \tag{3}$$

3.2 LoDiff-Visual vs. HiDiff-Policy.

Both models follow the same diffusion formulation, but differ in their inputs and configurations: **LoDiff-Visual** predicts future visual latents $\hat{\mathbf{v}}_{1:T}$ from an initial frame $\mathbf{v}_0$ and language instruction $\mathbf{l}$, using large spatial feature maps and a 1000-step diffusion schedule:

$$\hat{\mathbf{v}}_{1:T} = \text{LoDiff-Visual}(\mathbf{v}_0, \mathbf{l}), \tag{4}$$

HiDiff-Policy generates action sequences $\hat{\mathbf{a}}_{1:T}$ from the imagined video rollout $\hat{\mathbf{v}}_{1:T}$, using lower-dimensional inputs and a shorter 100-step schedule:

$$\hat{\mathbf{a}}_{1:T} = \text{HiDiff-Policy}(\hat{\mathbf{v}}_{1:T}), \tag{5}$$

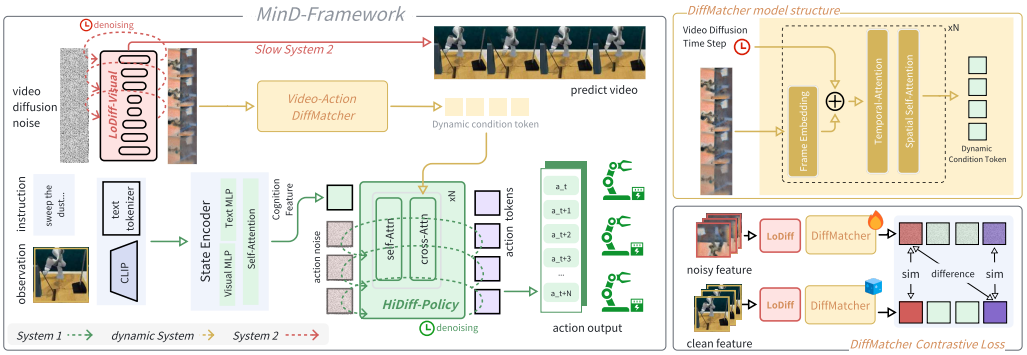

Figure 2: We present MinD, planning the robot action by using the video generation as the backbone. We pretrained the foundation model and the adapter on multiple robot pretraining datasets and finetuned them on downstream tasks, including RL-Bench simulation and a real-world Franka robot.

4 MinD: Manipulation in Dream

We present MinD, a scalable and general-purpose multimodal world model tailored for manipulation tasks in robotics. MinD jointly imagines future visual scenes and predicts coherent action sequences by decoupling video generation and action planning, while maintaining synchronized optimization between them. This section details the architecture, training paradigm of MinD.

4.1 Hierarchical Diffusion-based World Model Framework

We first describe our novel hierarchical diffusion-based world model framework. Given an initial visual observation $\mathbf{v}_0$ and a language instruction $\mathbf{l}$, MinD aims to generate a high-frequency action sequence $\hat{\mathbf{a}}_{1:T}$ for manipulation, along with a sequence of latent visual predictions $\hat{\mathbf{v}}_{1:T}$ for long-horizon imagination. As illustrated in Figure 2, the framework consists of three core components that operate asynchronously in a fast-slow system hierarchy:

Video Generation (LoDiff-Visual): Given the initial observation $\mathbf{v}_0$ and the language instruction $\mathbf{l}$, the video generator synthesizes a sequence of future visual observations $\{\mathbf{v}_t\}_{t=1}^{T}$ at a low temporal frequency. It utilizes a latent diffusion model (LDM) to produce semantically rich, long-horizon visual predictions. This component constitutes the *slow system*, focusing on high-level imagination and contextual consistency.

Action Planning (HiDiff-Policy): The action planner takes the generated video rollout as input and predicts a sequence of executable actions $\{\mathbf{a}_t\}_{t=1}^{T}$ at high temporal frequency. Leveraging a high-frequency diffusion transformer (DiT), this *fast system* captures fine-grained action dynamics and ensures real-time responsiveness and control fidelity.

Video-Action DiffMatcher: To bridge the asynchronous generation between visual imagination and action planning, we introduce the *DiffMatcher*, a temporal alignment module. Given a latent video tensor (B, C, T, H, W) and a denosing timestep τ, DiffMatcher embeds each frame using a two-layer MLP, applies learnable positional and timestep embeddings, and uses a Transformer encoder to model temporal dependencies. The output, a sequence of temporally-aware visual tokens (B, T, D_{proj}), is projected and injected into the DiT-based action planner as conditional context, grounding action generation in spatial semantics and temporal dynamics.

4.2 Dual-schedular-System Inference and Training

To handle the asynchronous nature of the two subsystems, we let each system reserve its own time-step scheduler for training, which independently controls the denoising steps of the video and action modules. This allows each subsystem to update at its own denosing frequency, ensuring stable convergence across asynchronous temporal scales.

Inference Pipeline: During inference, the video generator may update any k steps to obtain dynamic video latent features to the action planner HiDiff-Policy, depending on task complexity. The visual generator (LoDiff) forward-simulates noisy visual latents $\mathbf{v}_\tau$ at an arbitrary denoising timestep τ:

$$\mathbf{v}_\tau = \text{LoDiff-Visual}(\mathbf{v}_\mathcal{T}, \tau) \tag{6}$$

where $\mathbf{v}_\mathcal{T}$ is the initial noise. DiffMatcher converts $\mathbf{v}_\tau$ into aligned features $\mathbf{z}_\tau$:

$$\mathbf{z}_\tau = \text{DiffMatcher}(\mathbf{v}_\tau, \tau). \tag{7}$$

These features condition the action planner (HiDiff) to generate actions $\{\mathbf{a}_t\}_{t=1}^T$:

$$\hat{\mathbf{a}}_{1:T} = \text{HiDiff-Policy}(\mathbf{z}_\tau). \tag{8}$$

This pipeline enables LoDiff to generate dynamic visual latents, while DiffMatcher ensures robust feature alignment for real-time action planning.

Dual-System Co-Training To jointly optimize the asynchronous video and action subsystems, we propose a dual-system co-training strategy that enables coordinated learning while preserving the independence of each temporal stream. Formally, the training objective consists of the following components:

- **Video Loss** $\mathcal{L}_{\text{video}}$: a v-prediction [29] loss applied to the predicted video frames, defined as a pixel-level ℓ_2 loss , capturing both low-level fidelity and high-level semantics.
- **Action Loss** $\mathcal{L}_{\text{action}}$: a trajectory prediction loss that penalizes deviation from ground-truth actions, implemented as an ℓ_2 norm over joint angles or end-effector positions.

4.3 DiffMatcher Training

To improve the robustness of the action planner when dealing with noisy or partially synthesized visual inputs, we propose a diffusion-forced feature adaptation mechanism. Specifically, for the DiffMatcher, we simulate inference-time noise conditions during training as follows:

- Intermediate visual features $\mathbf{f}_v$ are extracted from either ground-truth or generated video frames.
- Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2\mathbf{I})$ is added to $\mathbf{f}_v$ at a large diffusion timestep $\tau \sim \mathcal{U}(T_{\text{min}}, T_{\text{max}})$, simulating partially denoised representations.
- A regularization loss $\mathcal{L}_{\text{sim}}$ is applied to enforce consistency between noisy features $\tilde{\mathbf{f}}_v$ and clean features $\mathbf{f}_v$ as shown in Eq. (9), where $\phi(\cdot)$ is the DiffMatcher.

$$\mathcal{L}_{\text{sim}} = \left\| \phi(\tilde{\mathbf{f}}_v) - \phi(\mathbf{f}_v) \right\|_2^2, \tag{9}$$

This method trains the adapter to produce noise-invariant representations, enhancing the action planner's ability to handle imperfect inputs and improving overall generalization.

4.4 Overall Training Objective

Therefore, the full training objective combines all loss components into a weighted sum:

$$\mathcal{L}_{\text{total}} = \lambda_1 \mathcal{L}_{\text{video}} + \lambda_2 \mathcal{L}_{\text{action}} + \lambda_3 \mathcal{L}_{\text{sim}}, \tag{10}$$

The hyperparameters $\lambda_1, \lambda_2, \lambda_3$ are set to balance reconstruction quality, control accuracy, modality alignment, and robustness. This co-training setup ensures that both the slow (LoDiff-Visual) and fast (HiDiff-Policy) systems learn in a mutually beneficial manner, while being robust to the inherent asynchrony and uncertainty in multimodal generation.

5 Experiments

In this section, we evaluate the performance of our proposed MinD framework through extensive experiments in both real-world and simulation environments. These experiments focus on manipulation tasks, comparing MinD with existing VLA models across multiple benchmarks.

Table 1: We compare the performance of MinD with existing VLA models across 7 tasks in RLBench settings. For the model using Mamba or LLM as the backbone, we colored it with a green background. We use a yellow background for the VLA models with a video generation backbone, and a red background for our method.

Methods	Pretrain modality	Backbone	Action Head	Phone on base	Toilet seat down	Close box
OpenVLA [18] (CoRL'24)	V+A	MLP	LLaMA2-7B	20%	76%	72%
CogAct-S [19] (Arxiv'24)	V+A	Diffusion	LLaMA2-7B	52%	44%	84%
CogAct-L [19] (Arxiv'24)	V+A	Diffusion	LLaMA2-7B	56%	100%	64%
RoboMamba [23] (NeruIPS'24)	V+A	MLP	Mamba-2.8B	44%	64%	60%
RoboDreamer [38] (ICML'24)	V	MLP	Video Diffusion-400M	64%	32%	88%
RoboDreamer-faster [38] (ICML'24)	V	MLP	Video Diffusion-400M	16%	4%	64%
MinD-S	V	Diffusion	Dc-UNet-1.5B	72%	48%	84%
MinD-B	V	Diffusion	Dc-UNet-1.5B	60%	32%	100%

Methods	put rubish in bin	Take umbrella out of stand	Close laptop lid	Sweep to dustpan	Mean Acc.% ↑	FPS (Hz) ↑
OpenVLA [18] (CoRL'24)	8%	28%	64%	68%	48.0%	6.3
CogAct-S [19] (Arxiv'24)	44%	40%	60%	64%	55.4%	9.6
CogAct-L [19] (Arxiv'24)	60%	32%	76%	44%	61.7%	8.6
RoboMamba [23] (NeruIPS'24)	36%	32%	52%	32%	45.7%	-
RoboDreamer [38] (ICML'24)	32%	40%	68%	76%	50.3%	0.7
RoboDreamer-faster [38] (ICML'24)	24%	20%	60%	64%	37.4%	1.1
MinD-S	32%	24%	60%	84%	58.0%(-2.4%)	11.3
MinD-B	52%	32%	68%	96%	***63.0%***($\pm 1.5\%$)	10.2

5.1 Model Implementation Details

We implemented our model based on the publicly available Dynamicrafter framework [34], which serves as the pretrained visual backbone. The core architecture of Dynamicrafter is preserved, including its visual encoder and spatial-temporal UNet backbone. For the robot task training, we follow the setting of OpenVLA [18]. The state feature for HiDiff-Policy uses the same CLIP and text tokenizer as Dynamicrafter, further reducing the inference time of feature encoding as shown in Fig. 2. For the HiDiff-Policy module, we implement two versions with different parameter sizes: *Small* and *Base*, following the same action policy model size as CogACT[19], but with extra cross-attention layers. To ensure reproducibility, we used the same data preprocessing pipeline as Dynamicrafter, including image resolution resizing to 128x128 and text tokenizer from CLIP [27]. We follow the setting of OpenVLA to preprocess our robot datasets, using RLDS [28] format action data. All hyperparameters and implementation details will be released with our codebase.

Pretraining We use RT-1 [3] for feature adapter alignment pretraining. RT-1 includes 87k dense trajectories, and 128*128 for both HiDiff-Policy and LoDiff-Visual training. All experiments were conducted using PyTorch 2.5.1 with CUDA 12.1 support. The training is performed on 4 NVIDIA A100 GPUs, with a batch size of 16 and an initial learning rate of 2e-5. We used the AdamW optimizer with a cosine learning rate schedule and a warmup of 2000 steps. The model is trained for 50k steps on RT-1 [3] for pretraining and feature alignment, and fine-tuned on the downstream class. The detailed fine-tuning setups will be included in the following sections.

5.2 Manipulation in Silumation

Simulation Benchmark Setting We use RL-Bench [17] in the CoppeliaSim simulator as our simulation benchmark. The RL-Bench uses a Franka Panda robot, with multiview cameras, while we collect front-view images as our input. We include 7 tasks for comparison. We use the predefined waypoints given by RL-Bench, and downsample the dense frames to key-frames similar to previous works [11, 36] to construct the fine-tuning dataset. We collect 1000 trajectories in total, including 100 trajectories for each 7 tasks, and 300 more randomly sampled tasks for improving the robustness.

Fine-tuning and Evaluation Details We fine-tune the pretrained checkpoint on the RL-Bench fine-tuning dataset. The training process is conducted using the Adam optimizer with a learning rate of $2e-5$, and the model is trained for 10k steps. The input image resolution is set to 128×128, and data augmentation techniques such as random cropping, flipping, and color jittering were applied to improve generalization. Evaluation is performed using the success rate for each task, averaged over

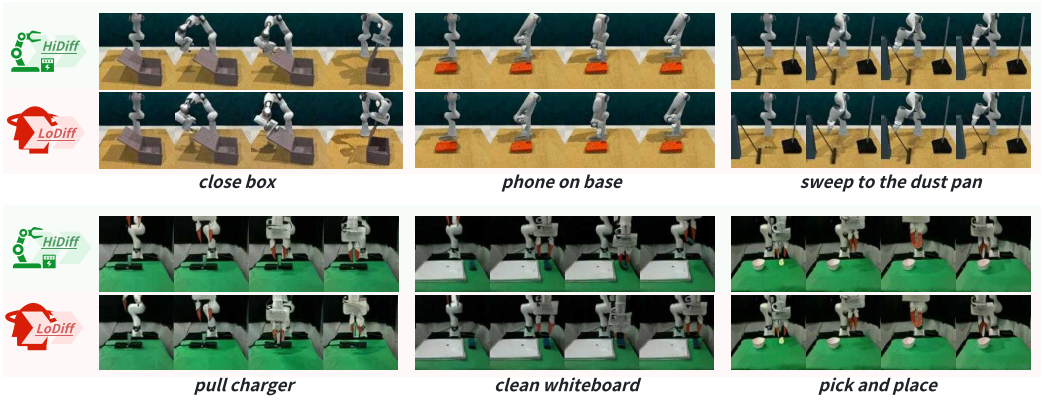

Figure 3: Qualitative comparison of video generation result from LoDiff-Visual against real execution observation of HiDiff-Policy of RL-Bench and real-world Franka.

25 attempts per task. We implement two versions of MinD with different HiDiff-Policy model sizes, the detailed model information will be included in the Appendix.

Baselines We compare our model with several VLA baselines, including RoboDreamer [38], OpenVLA [18], and CogACT [19]. These baselines employ different backbone architectures and training paradigms, such as transformer-based models or convolutional neural networks. We use the same RL-Bench tasks and evaluation metrics for all methods.

Inference Speed We measure the average step prediction time on an RTX-4090 GPU. Notably, MinD-S achieves the highest inference speed of **11.3 FPS**, showcasing its superior efficiency. In contrast, the video generation-based world model RoboDreamer is limited to an inference speed of approximately 1 FPS. While reducing its video generation steps (e.g., decreasing DDIM steps from 100 to 50) slightly increases FPS, it significantly degrades the success rate. This highlights the effectiveness of our proposed DiffMatcher and dual-schedule design.

Quantitative Results Table 1 shows that our models, MinD-S and MinD-B, achieve state-of-the-art performance on RLBench tasks among three different types of VLA models. MinD-B achieves the highest mean accuracy of **63.0%**, while MinD-S follows with **58.0%**, outperforming all baseline methods. Notably, our models excel in tasks requiring complex temporal reasoning, such as "Sweep to Dustpan" (**96%**) and "Close Laptop Lid" (**68%**). These results highlight that video generation models can serve as a strong foundational backbone for comprehensive visual-language manipulations.

5.3 Manipulation in Real-world Robotics

Real-World Franka Robot Setup We use a Franka Research 3 robot for the experiment. The robot is set up with a front-view camera. We work on 4 tasks: *1) pick and place, 2) unplug the charger, 3) pour water, 4) wipe the whiteboard* for training data collection and testing. For each task, we collect 100 human demonstration trajectories via teleoperation using a SpaceMouse.

Training and Evaluation Details We fine-tuned the pretrained checkpoint on the self-collected fine-tuning dataset. The fine-tuning process and parameter setups follow the RL-Bench experiments as described in 5.2. For each single task, we use the latest checkpoints and evaluate 20 times. Both training and evaluation processes are conducted in the same robotic setup to ensure consistency between data collection and testing conditions.

Quantitative Results Table 2 summarizes the results, comparing our method against other baselines. As shown, our method achieves competitive performance across all tasks, with notable strengths in tasks requiring precise manipulation, such as *pick and place*(60%) and *wiping the whiteboard* (65%). The 50% average success rate across tasks demonstrates the robustness of our approach.

Table 2: Real-world Franka robot execution success rate comparison across four tasks: *pick and place*, *unplug the charger*, *pour water*, and *wipe the whiteboard*. Results are shown for two methods: OpenVLA and MinD, with the average success rate reported for each method. Under limited training data, MinD outperforms the OpenVLA.

Methods	Pick&Place	Unplug charger	Pour water	Wipe blackboard	Average
OpenVLA	40%	35%	**45%**	50%	42.5%
DiffusionPolicy	55%	**45%**	30%	45%	43.8%
MinD	**60%**	40%	35%	**65%**	***50.0%***

Table 3: Ablation study of MinD across different configurations of modalities (A: action, V: video) and trainable modules. SE denotes the state encoder, and LDP represents large-scale data pretraining. We evaluate each configuration based on video generation quality (FVD [30]) and success rate (SR) in task execution. The results highlight the impact of key components such as LDP, diffusion modules (LoDiff, DiffMatcher, HiDiff), and loss functions ($\mathcal{L}_{\text{video}}$, $\mathcal{L}_{\text{sim}}$, $\mathcal{L}_{\text{action}}$) on performance.

modality	trainable module					Loss			FVD ↓	SR ↑
	LDP	LoDiff	DiffMatcher	HiDiff	SE	$\mathcal{L}_{\text{video}}$	$\mathcal{L}_{\text{sim}}$	$\mathcal{L}_{\text{action}}$		
A	-	-	✓	✓	-	-	-	✓	–	44.0%
V	✓	✓	-	-	-	✓	-	-	**235.5**	-
V	-	✓	-	-	-	✓	-	-	352.6	-
A+V		✓	✓	✓	✓	✓	✓	✓	393.3	55.4%
A+V	✓	✓	✓	✓	✓	-	✓	✓	378.3	**64.0%**
A+V	✓	✓	✓	✓	✓	-	-	✓	596.5	58.3%
A+V	✓	✓	✓	✓	✓	✓	✓	✓	307.1	63.4%

5.4 Ablation Study

The results of the ablation study are presented in Table 3. Each row corresponds to a different configuration of MinD, with variations in modalities (action-only, video-only, or both) and trainable modules. The evaluation metrics include the Fréchet Video Distance (FVD), which measures the quality of generated videos, and the success rate (SR), which quantifies task execution performance.

1. **Large scale video data Pretraining (LDP)**: The inclusion of large-scale data pretraining significantly improves both FVD and SR. For instance, the configuration with A+V, LDP, and all loss terms achieves the highest SR (64.0%) while maintaining competitive FVD (378.3).

2. **Diffusion Modules**: The combination of LoDiff, DiffMatcher, and HiDiff plays a crucial role in improving task execution success. Configurations without these modules (e.g., rows with "-" for LoDiff or DiffMatcher) show a marked drop in performance.

3. **Loss Functions**: The ablation of specific loss terms also impacts performance. For example, removing $\mathcal{L}_{\text{video}}$ and $\mathcal{L}_{\text{sim}}$ (last row) leads to a lower SR (58.3%) and a significantly higher FVD (596.5), indicating the importance of these loss terms for aligning video generation with task requirements.

4. **Preservation of VLA and World Model**: The inclusion of both $\mathcal{L}_{\text{video}}$ and $\mathcal{L}_{\text{sim}}$ ensures that the system retains the functionalities of manipulation(SR 63.4%) and the video prediction(FVD 307.1). Ablating these losses, as seen in the last row, leads to a drop in SR (58.3%) and a sharp increase in FVD (596.5), indicating degraded performance in both video generation and task execution.

Overall, the results demonstrate that incorporating both modalities, pretraining, and a full set of loss terms is critical for achieving optimal task execution success and video generation quality. These findings provide insights into the design of future multimodal frameworks for robotic learning.

6 Case Study: Can Video Generation Enable Trustworthy VLA?

World-model-based visual-language alignment (VLA) aims to enhance interpretability and decision-making. Here, we investigate whether video generation models (VGMs) can enable trustworthy VLA by predicting task outcomes and identifying potential risks. Using RLBench and the Franka robot across 32 cases, we demonstrate how VGMs provide actionable insights for safer execution.

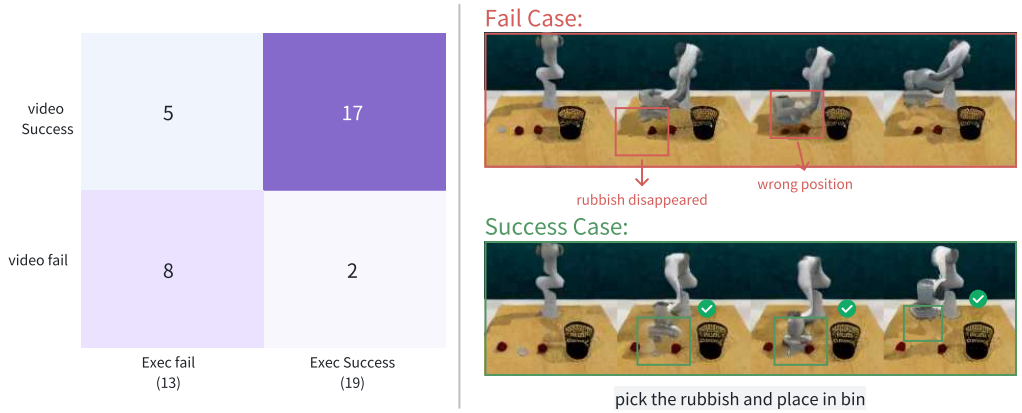

Figure 4: Evaluation of video generation predictions. The left panel shows the confusion matrix, highlighting prediction accuracy for task outcomes. The right panel visualizes a failing case (top) with trajectory misalignment and a successful case (bottom) with accurate prediction.

Video Generation for Risk Assessment. As shown in Fig. 4, VGMs align video predictions with real-world outcomes. Among 19 successful executions, 17 are correctly labeled (89.4% true positive rate). For 13 failed cases, 8 are accurately predicted (61.53% true negative rate). These results suggest VGMs can serve as a proxy for risk assessment, allowing systems to anticipate and mitigate potential failures before execution.

Failure Case Insights. The right panel of Fig. 4 highlights two scenarios. The top row shows a failure where the model predicts success, but trajectory misalignment causes execution to fail. The bottom row illustrates a success where predicted and real trajectories align. These examples show VGMs' potential to predict outcomes while exposing limitations in capturing complex dynamics. More success cases are included in Fig. 3, where the execution success is often achieved together with higher quality video prediction.

Toward Safer and More Trustworthy VLA. VGMs provide a foundation for trustworthy VLA by simulating and analyzing potential actions. This enables systems to identify risks, refine plans, and enhance safety. Future work should focus on improving motion prediction and integrating multimodal inputs to further boost reliability and robustness.

7 Conclusion

In this work, we present MinD, a hierarchical diffusion-based world model designed for VLA tasks. Through extensive experiments, we demonstrate that MinD serves as a scalable and effective backbone for VLA, addressing key challenges such as slow generation speed and poor video-action consistency. Our framework introduces a Fast-Slow Coupled Architecture, which allows for efficient video generation and real-time action planning, and a novel DiffMatcher module, which bridges the gap between imagined videos and executable actions. Additionally, we conduct a trustworthy analysis of MinD, showing its potential to predict task outcomes and proactively mitigate risks, making VGMs a promising backbone for VLA systems. These results highlight the unique advantages of video generation models in enabling robust and interpretable VLA, advancing their utility in robotics.

Limitation Despite its advancements, MinD is still constrained by the availability and diversity of robot training data, which limits its ability to generalize from arbitrary video inputs to robotic actions. The reliance on domain-specific datasets and task configurations hinders the model's scalability to more general-purpose video-to-robot applications. Future work should explore methods to enhance data diversity, incorporate large-scale multimodal pretraining, and improve generalization across unseen tasks and environments. These efforts are essential for building an open-world world model that can generalize beyond robotic datasets to broader, real-world scenarios.

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] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. pi0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[3] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

[4] Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, et al. Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024.

[5] 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.

[6] Haoxin Chen, Yong Zhang, Xiaodong Cun, Menghan Xia, Xintao Wang, Chao Weng, and Ying Shan. Videocrafter2: Overcoming data limitations for high-quality video diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7310–7320, 2024.

[7] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, page 02783649241273668, 2023.

[8] Xiaowei Chi, Hengyuan Zhang, Chun-Kai Fan, Xingqun Qi, Rongyu Zhang, Anthony Chen, Chi-min Chan, Wei Xue, Wenhan Luo, Shanghang Zhang, et al. Eva: An embodied world model for future video anticipation. *arXiv preprint arXiv:2410.15461*, 2024.

[9] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, et al. Palm-e: An embodied multimodal language model. 2023.

[10] Yilun Du, Mengjiao Yang, Pete Florence, Fei Xia, Ayzaan Wahid, Brian Ichter, Pierre Sermanet, Tianhe Yu, Pieter Abbeel, Joshua B Tenenbaum, Leslie Kaelbling, Andy Zeng, and Jonathan Tompson. Video language planning. *arXiv [cs.CV]*, October 2023.

[11] Ankit Goyal, Jie Xu, Yijie Guo, Valts Blukis, Yu-Wei Chao, and Dieter Fox. Rvt: Robotic view transformer for 3d object manipulation. In *Conference on Robot Learning*, pages 694–710. PMLR, 2023.

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

[13] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

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

[15] Yucheng Hu, Yanjiang Guo, Pengchao Wang, Xiaoyu Chen, Yen-Jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. Video prediction policy: A generalist robot policy with predictive visual representations. *arXiv preprint arXiv:2412.14803*, 2024.

[16] Siyuan Huang, Liliang Chen, Pengfei Zhou, Shengcong Chen, Zhengkai Jiang, Yue Hu, Yue Liao, Peng Gao, Hongsheng Li, Maoqing Yao, et al. Enerverse: Envisioning embodied future space for robotics manipulation. *arXiv preprint arXiv:2501.01895*, 2025.

[17] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2):3019–3026, 2020.

[18] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model, 2024.

[19] Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, Xiaofan Wang, Bei Liu, Jianlong Fu, Jianmin Bao, Dong Chen, Yuanchun Shi, Jiaolong Yang, and Baining Guo. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation, 2024.

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

[21] Ying Li, Xiaobao Wei, Xiaowei Chi, Yuming Li, Zhongyu Zhao, Hao Wang, Ningning Ma, Ming Lu, and Shanghang Zhang. Manipdreamer: Boosting robotic manipulation world model with action tree and visual guidance. *arXiv preprint arXiv:2504.16464*, 2025.

[22] Jiaming Liu, Hao Chen, Pengju An, Zhuoyang Liu, Renrui Zhang, Chenyang Gu, Xiaoqi Li, Ziyu Guo, Sixiang Chen, Mengzhen Liu, et al. Hybridvla: Collaborative diffusion and autoregression in a unified vision-language-action model. *arXiv preprint arXiv:2503.10631*, 2025.

[23] Jiaming Liu, Mengzhen Liu, Zhenyu Wang, Pengju An, Xiaoqi Li, Kaichen Zhou, Senqiao Yang, Renrui Zhang, Yandong Guo, and Shanghang Zhang. Robomamba: Efficient vision-language-action model for robotic reasoning and manipulation, 2024.

[24] Fuchen Long, Zhaofan Qiu, Ting Yao, and Tao Mei. Videodrafter: Content-consistent multi-scene video generation with llm. *arXiv preprint arXiv:2401.01256*, 2024.

[25] OpenAI. Video generation models as world simulators. Technical report, OpenAI, 2024.

[26] William Peebles and Saining Xie. Scalable diffusion models with transformers. *arXiv preprint arXiv:2212.09748*, 2022.

[27] 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.

[28] Sabela Ramos, Sertan Girgin, Léonard Hussenot, Damien Vincent, Hanna Yakubovich, Daniel Toyama, Anita Gergely, Piotr Stanczyk, Raphael Marinier, Jeremiah Harmsen, Olivier Pietquin, and Nikola Momchev. Rlds: an ecosystem to generate, share and use datasets in reinforcement learning, 2021.

[29] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

[30] Thomas Unterthiner, Sjoerd Van Steenkiste, Karol Kurach, Raphaël Marinier, Marcin Michalski, and Sylvain Gelly. Fvd: A new metric for video generation. 2019.

[31] Junjie Wen, Yichen Zhu, Jinming Li, Minjie Zhu, Zhibin Tang, Kun Wu, Zhiyuan Xu, Ning Liu, Ran Cheng, Chaomin Shen, et al. Tinyvla: Towards fast, data-efficient vision-language-action models for robotic manipulation. *IEEE Robotics and Automation Letters*, 2025.

[32] Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. Unleashing large-scale video generative pre-training for visual robot manipulation. In *International Conference on Learning Representations*, 2024.

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

[34] Jinbo Xing, Menghan Xia, Yong Zhang, Haoxin Chen, Xintao Wang, Tien-Tsin Wong, and Ying Shan. Dynamicrafter: Animating open-domain images with video diffusion priors. *arXiv preprint arXiv:2310.12190*, 2023.

[35] Mengjiao Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. *arXiv preprint arXiv:2310.06114*, 2023.

[36] Rongyu Zhang, Menghang Dong, Yuan Zhang, Liang Heng, Xiaowei Chi, Gaole Dai, Li Du, Yuan Du, and Shanghang Zhang. Mole-vla: Dynamic layer-skipping vision language action model via mixture-of-layers for efficient robot manipulation. *arXiv preprint arXiv:2503.20384*, 2025.

[37] Haoyu Zhen, Qiao Sun, Hongxin Zhang, Junyan Li, Siyuan Zhou, Yilun Du, and Chuang Gan. Tesseract: learning 4d embodied world models. *arXiv preprint arXiv:2504.20995*, 2025.

[38] Siyuan Zhou, Yilun Du, Jiaben Chen, Yandong Li, Dit-Yan Yeung, and Chuang Gan. RoboDreamer: Learning compositional world models for robot imagination. *arXiv [cs.RO]*, April 2024.

A Extensive Experiments

A.1 Wrist View Video Generation and Manipulation

We further evaluate our method's ability to generate and utilize wrist-view videos for real-world robotic manipulation tasks. As shown in Fig. 5, the generated wrist view videos maintain high visual fidelity and temporal coherence, especially in tasks involving object interaction such as *Pick&Place* and *Unplug Charger*. These videos serve not only as realistic reconstructions but also as effective guidance in the feature space, enabling better policy learning and decision-making.

To validate the utility of wrist view generation in real-world deployment, we conduct robot execution experiments on a Franka Panda arm across the two aforementioned tasks. We compare two settings: using the wrist view versus the front view during policy training and inference. The execution success rates are summarized in Table 4.

Table 4: Real-world Franka robot execution success rate comparison across two tasks: *Pick&Place* and *Unplug Charger*.

Method	View	Pick&Place	Unplug Charger	Average
MinD	Wrist	**60%**	80%	***70.0%***
MinD	Front	**60%**	40%	***50.0%***

The results demonstrate that incorporating wrist view generation significantly improves execution success, particularly for fine-grained manipulation. In the *Unplug Charger* task, the wrist view offers better perception of contact points and spatial relations, achieving a 40% improvement over the front view baseline.

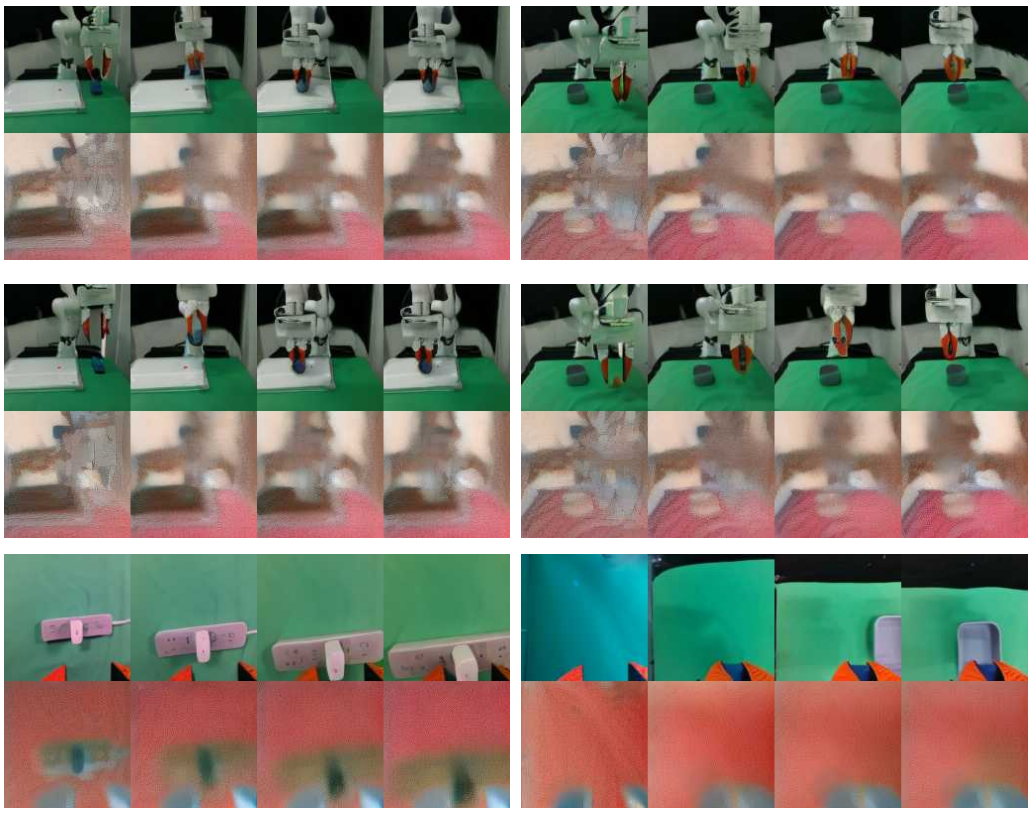

Figure 5: More visualization results of the video generation in feature level or DDIM=100. The first two rows are the front view, last row is the wrist view. We noticed that the wrist view could significantly reduce the strict pre-setting of the camera pose and position initialization.

A.2 More visualization results

While wrist view introduces challenges such as rapid scene shifts, it reduces the need for complex setups like external camera calibration. It naturally provides precise relative position cues between the gripper and the object, enabling better generalization in generation models. This leads to tighter alignment between video generation and control, making the output more actionable and transferable to real-world manipulation.

Figure 6: Qualitative comparison of video prediction ability with the real-execution results.

Moreover, we shows the feature of the video generation model under different loss settings Fig. 6. If we do not preprocess the RLDS dataset into video format, and directly use the image loss, the video model would become a fixed frame model without temporal information. While the task successful rate still remains better than ordinary diffusion policy without temporal cross-attention insertion. If, without any video loss, the video feature would become noisy, as shown in the second figure. Using the video loss would help the MinD to retain its video generation ability.

A.3 Failing case Prediction

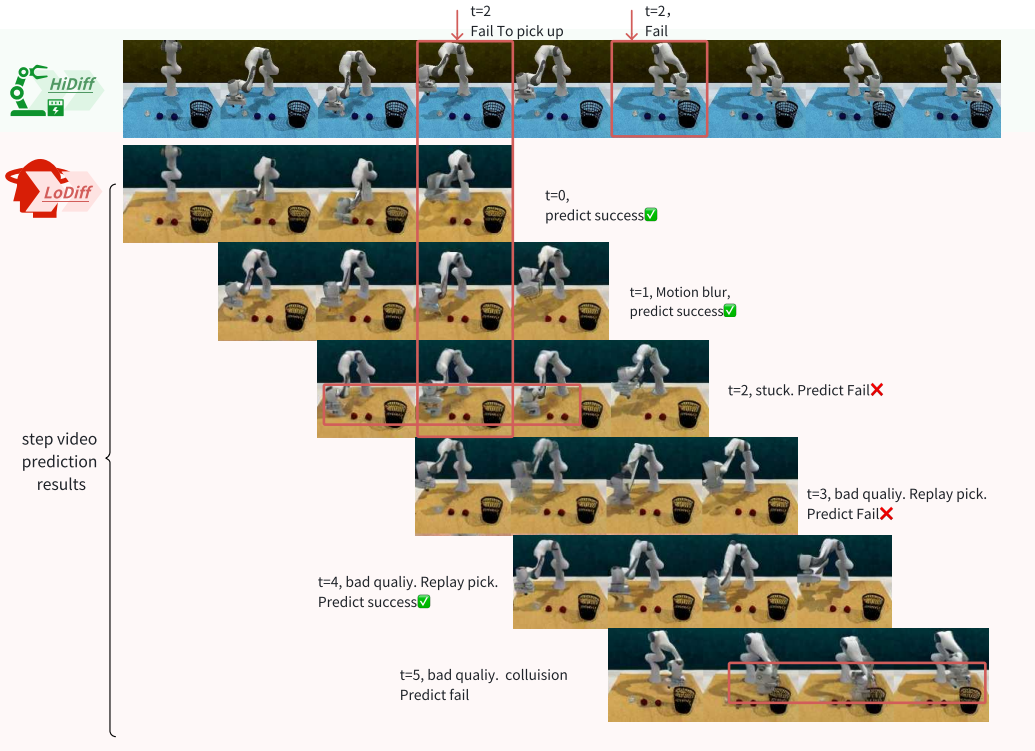

Figure 7: Qualitative comparison of video prediction ability with the real-execution results.

Fig. 7illustrates a qualitative comparison between real-world robot execution results (HiDiff) and predicted video rollouts (LoDiff) for manipulation tasks. While HiDiff shows the actual outcomes—such as missed grasps, obstruction, or collisions—LoDiff predicts these failures in advance.

At t = 2, for example, the robot in HiDiff fails to pick up the object. LoDiff, despite being a forward video prediction model, successfully forecasts this failure by showing the gripper getting stuck or misaligned even before the actual failure occurs. Similarly, LoDiff captures issues such as motion blur, replayed picking actions, and eventual collisions in later steps (t = 3 to t = 5), aligning with the error patterns observed in HiDiff.

This demonstrates that LoDiff can serve as an early warning signal, allowing the system to anticipate and potentially avoid failure before real-world execution, by detecting subtle cues like object shifts, bad grasps, or trajectory deviations.

B Detailded Model Parameters

B.1 Video Generation Model

Our video generation framework is built upon the Latent Visual Diffusion Model (LVDM) architecture, based on Dynamicrafter [34], with adaptations to suit the low-resolution setting of 128×128 and short video clips of 4 frames. The model adopts a denoising diffusion process with 1000 steps and a linear noise schedule ranging from $\beta = 0.00085$ to $\beta = 0.012$. We use a velocity-based parameterization and enable zero-SNR rescaling to stabilize training.

The generative model consists of three main components: a 3D U-Net backbone, a latent-space autoencoder, and a hybrid conditioning mechanism that incorporates both textual and visual inputs. The U-Net is configured with 4 spatial resolution levels and channel multipliers {1, 2, 4, 4}, starting from a base width of 320. It applies spatial attention at resolutions {4, 2, 1} and temporal attention over 4-frame sequences. Each level contains two residual blocks, and attention heads have 64 channels. Temporal reasoning is handled via temporal convolution and attention mechanisms, and the model supports both image and text cross-attention. The first-stage autoencoder encodes RGB video

Table 5: Video Generation Model Architecture Overview

Component	Configuration
Input Resolution	128×128, 4 frames
Latent Space	$16 \times 16 \times 4$ (spatial $\times$ temporal), 4 channels
Diffusion Steps	1000
Noise Schedule	Linear ($\beta \in [0.00085, 0.012]$)
Parameterization	Velocity-based
Conditioning	Hybrid (Text + Image)
Text Encoder	Frozen OpenCLIP (penultimate layer)
Image Encoder	Frozen OpenCLIP Image Embedder
Image Projector	Transformer Resampler (4 layers, 12 heads, 1024 dim)

frames into a latent space of size $16 \times 16 \times 4$ (spatial $\times$ temporal), with 4 channels. The encoder uses a ResNet-style convolutional structure with 4 downsampling levels, each with 2 residual blocks. Attention is not used in the autoencoder, and reconstruction is performed using a symmetric decoder.

For conditioning, a hybrid strategy is adopted. Textual inputs are encoded using a frozen OpenCLIP model (penultimate layer), while visual inputs are processed using a frozen OpenCLIP image encoder followed by a trainable Resampler module. The Resampler is a Transformer-based attention module with 4 layers, 12 heads (each of width 64), and 16 learnable queries. It projects 1280-dimensional OpenCLIP features into a 1024-dimensional conditioning vector that aligns with the U-Net's context embedding space.

Training is performed on a dataset of 4-frame video clips with spatial resolution 128×128, sampled at a frame stride of 4. We use a batch size of 8 and train with mixed precision (FP16) across 4 GPUs. Gradients are accumulated every 2 steps and clipped to a norm value of 0.5. The model is initialized from a pretrained checkpoint (`epoch=53-step=12000.ckpt`) and trained with a base learning rate of 2×10^{-5}. During training, image samples are periodically generated using DDIM sampling with 50 steps and a guidance scale of 7.5 for qualitative evaluation.

A summary of the model architecture and key parameters is provided in Table 5 and Table 6.

Table 6: Detailed Configuration of U-Net and Autoencoder

Module	Parameter	Value
U-Net	Base Channels	320
	Channel Multipliers	{1, 2, 4, 4}
	Attention Resolutions	{4, 2, 1}
	Transformer Depth	1
	Head Channels	64
	Temporal Length	4
Autoencoder	Input Channels	3
	Embedding Dim	4
	Resolution	128×128
	Channel Multipliers	{1, 2, 4, 4}
	ResBlocks per Level	2

B.2 DiT-based Action Model Architecture

Our action model is based on the Diffusion Transformer (DiT) architecture [26], originally designed for image generation tasks, and we extend it for temporally-conditioned action prediction in robotics. The model is adapted to predict future robot actions using a diffusion process, with cross-attention to visual features and classifier-free guidance.

Architecture Overview. The action model receives as input: (1) a noisy action sequence $x \in \mathbb{R}^{N \times T \times D}$, (2) a diffusion timestep t, and (3) condition embeddings z, including both task embeddings and visual features. The architecture consists of four key components:

Action Embedder: A linear projection maps the input action sequence to the transformer hidden space.

Timestep Embedder: A sinusoidal MLP-based embedding module encodes the scalar timestep t into a vector.

Condition Embedder: A $LabelEmbedder$ projects task condition features and applies stochastic token dropping to support classifier-free guidance [14].

Transformer Backbone: A stack of L DiT blocks (each with self-attention, optional cross-attention, and MLP sublayers) processes the embedded sequence.

Cross-Attention with Video Features. Each DiT block optionally includes a cross-attention mechanism that injects video features (extracted from a pretrained encoder) as conditioning. A residual gate balances the contribution of cross-attention dynamically:

$$x \leftarrow x + \tanh(\gamma) \cdot \text{CrossAttn}(\text{LN}(x), \text{video\_features})$$

where γ is a learnable gating parameter initialized to 1.

Prediction Head. The final prediction is computed via two linear layers:

- One for the action outputs (e.g., joint positions or velocities),

- One for the gripper state (binary open/close), processed separately to improve stability.

These outputs are concatenated and returned as the model prediction for each timestep.

Classifier-Free Guidance. To enable conditional generation control, we implement classifier-free guidance by duplicating the batch and replacing condition tokens with a learned null token in the unconditional path. At inference, both conditional and unconditional predictions are combined:

$$\epsilon_{\text{cfg}} = \epsilon_{\text{uncond}} + s \cdot (\epsilon_{\text{cond}} - \epsilon_{\text{uncond}})$$

where s is the guidance scale.

Model Variants. We consider two model variants:

- **DiT-Small (DiT-S)**: 6 layers, hidden size 384, 4 attention heads.
- **DiT-Base (DiT-B)**: 12 layers, hidden size 768, 12 attention heads.

An overview of the configuration is provided in Table 7.

Table 7: DiT Action Model Configuration

Model	Depth	Hidden Size	Heads
DiT-S	6	384	4
DiT-B	12	768	12

Loss Function. The model is trained using denoising score matching. Given predicted noise $\hat{\epsilon}$ and true noise ϵ added to the actions, the objective is:

$$\mathcal{L}_{\text{MSE}} = \mathbb{E}\left[\|\hat{\epsilon} - \epsilon\|^2\right]$$

This aligns with the standard diffusion model training paradigm, adapted for continuous-valued action spaces.

C RLBench Evaluation Setup

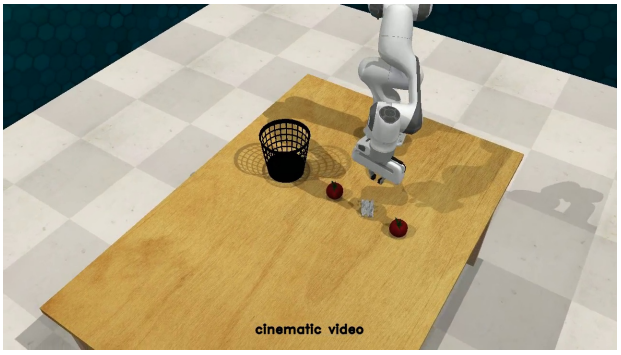

Figure 8: Rlbench Experiment Setups

To evaluate our method in simulation, we adopt the RLBench benchmark [**?**], a large-scale vision-based robot learning environment built on CoppeliaSim. We follow the common protocol for few-shot imitation tasks and evaluate our framework across multiple challenging manipulation scenarios.

Simulator Environment. We use the official RLBench Python API with $JointVelocity + DiscreteGripper$ as the action mode and $FrontRGB$ camera view at a resolution of 224×224. The environment is initialized in headless mode for efficient parallel evaluation. Each episode is capped at 15 steps, and success is determined by environment-defined termination signals or task-specific reward triggers.

Evaluated Tasks. We select a diverse subset of 7 RLBench tasks to evaluate the generalization and robustness of our model:

- `close_box`
- `toilet_seat_down`
- `sweep_to_dustpan`
- `put_rubbish_in_bin`
- `phone_on_base`
- `take_umbrella_out_of_umbrella_stand`

- `close_laptop_lid`

These tasks are carefully chosen to cover diverse skill requirements, including object manipulation, precise placement, and long-horizon planning.

Model Evaluation Protocol. For each task, we run 25 test episodes using the $predict$ mode, where the model generates actions given language instruction and visual observation. Our policy model is based on a DiT-B transformer backbone trained with our collaborative diffusion-autoregressive mechanism.

During each episode, the model receives observations from the front camera and the current robot state. It then predicts the next action using the $predict_action$ interface, which internally fuses multi-frame visual information. The resulting action is executed in the RLBench environment, and the outcome is recorded.

Execution Details. We parallelize task execution across 8 GPUs using $xvfb - run$ and $CUDA\_VISIBLE\_DEVICES$ to enable headless rendering and maximize throughput. Each task is mapped to a different GPU device for simultaneous evaluation. Video recordings of both predicted actions and cinematic rollouts are saved for post-analysis. Average inference time, environment step latency, and episode success rates are logged for each run.

Reproducibility. The evaluation script is implemented in Python and based on a wrapper around RLBench and our visual-language-action (VLA) model. All evaluations are deterministic given the same random seed and dataset, and our codebase supports both offline replay and online prediction modes. We provide full configuration files and checkpoints for reproducibility.

D Real-World Franka Setups

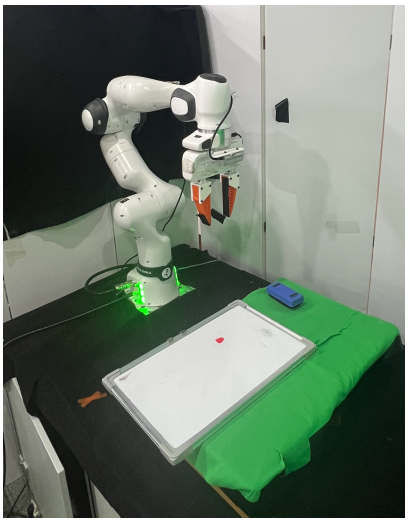

Figure 9: Franka Experiment Setups

Our real-world robotic experiments are conducted using the Franka Research 3 (FR3) robotic arm, following hardware and software configurations consistent with HybridVLA [22]. Specifically, we operate the FR3 with controller version 5.6.0, `libfranka` version 0.13.3, and Franka ROS interface version 0.10.0, under Ubuntu 20.04 and ROS Noetic. The robot is set to active execution mode with the FCI switch enabled, allowing low-level torque control and real-time trajectory tracking.

For visual observation, we utilize two Intel RealSense cameras: a front-view **RealSense D435** and a wrist-mounted **RealSense D515**, capturing synchronized RGB inputs at a resolution of 224×224. These inputs are paired with the robot state, which includes full 7-DoF end-effector poses: 3D translation ($\Delta x, \Delta y, \Delta z$), 3D Euler angles (Roll, Pitch, Yaw), and 1D gripper state ($g \in \{0, 1\}$), formulated as:

$$a = [\Delta x, \Delta y, \Delta z, \text{Roll}, \text{Pitch}, \text{Yaw}, g]$$

Training demonstrations are collected by teleoperation using a 3Dconnexion SpaceMouse, allowing users to perform tasks from various tabletop positions. Each task includes 100 demonstration trajectories, providing diverse motion patterns and object interactions.

We evaluate our model on the following single-arm real-world manipulation tasks, consistent with prior benchmarks [22]:

- **Pick and Place:** Pick a specifically described object and place it into a matching container.
- **Unplug Charger:** Grasp and remove a charger from a socket with appropriate rotation and lifting.
- **Pour Water:** Pick and tilt a bottle to pour its contents into a cup.

- **Wipe Blackboard:** Grasp an eraser and remove red markings from a whiteboard surface.

This setup ensures compatibility with other state-of-the-art VLA models such as Diffusion Policy [7] and CogACT [19], and supports robust benchmarking of language-conditioned visuomotor policies in real-world robotic manipulation.

E Code

Our demo page and code: https://manipulate-in-dream.github.io/