TesserAct: Learning 4D Embodied World Models

Haoyu Zhen[1] *   Qiao Sun[1] *   Hongxin Zhang[1]   Junyan Li[1]   Siyuan Zhou[2]
Yilun Du[3]   Chuang Gan[1]

[1]UMass Amherst   [2]HKUST   [3]Harvard University
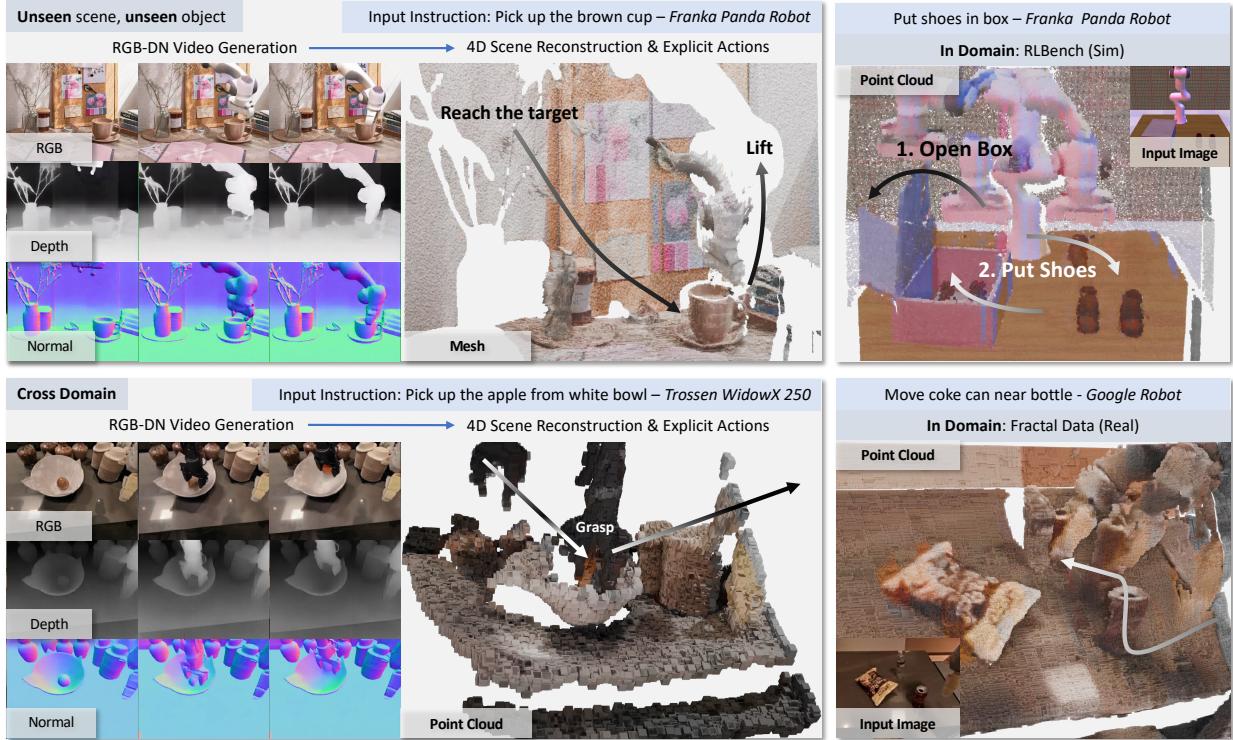
https://TesserActWorld.github.io

Figure 1. We propose TesserAct, the 4D Embodied World Model, which takes an input image and text instruction to generate RGB, depth, and normal videos, reconstructing a 4D scene and predicting actions. Our model not only achieves strong performance on in-domain data (right) but also generalizes effectively to unseen scenes, novel objects (top left), and cross-domain scenarios (bottom left).

Abstract

This paper presents an effective approach for learning novel 4D embodied world models, which predict the dynamic evolution of 3D scenes over time in response to an embodied agent's actions, providing both spatial and temporal consistency. We propose to learn a 4D world model by training on RGB-DN (RGB, Depth, and Normal) videos. This not only surpasses traditional 2D models by incorporating detailed shape, configuration, and temporal changes into their predictions, but also allows us to effectively learn accurate inverse dynamic models for an embodied agent. Specifically, we first extend existing robotic manipulation video datasets with depth and normal information leveraging off-the-shelf models. Next, we fine-tune a video generation model on this annotated dataset, which jointly predicts RGB-DN (RGB, Depth, and Normal) for each frame. We then present an algorithm to directly convert generated RGB, Depth, and Normal videos into a high-quality 4D scene of the world. Our method ensures temporal and spatial coherence in 4D scene predictions from embodied scenarios, enables novel view synthesis for embodied environments, and facilitates policy learning that significantly outperforms those derived from prior video-based world models.

---

*Equal contribution.

1. Introduction

Learned world models [21, 64, 67, 77], which simulate environmental dynamics, play a crucial role in enabling embodied intelligent agents. Such models enable flexible policy synthesis [15, 40], data simulation and generation [67, 80], and long-horizon planning [14, 28, 74]. However, while the physical world is three-dimensional in nature, existing world models operate in the space of 2D pixels. This limitation leads to an incomplete representation of spatial relationships, impeding tasks that require precise depth and pose information. For instance, without accurate depth and 6-DoF pose estimations, robotic systems struggle to determine the exact position and orientation of objects. Furthermore, existing 2D models can produce unrealistic results, such as inconsistent object sizes and shapes across time steps, which limits their use in data-driven simulations and robust policy learning.

In this paper, we explore how we can instead learn a 4D embodied world model, TesserAct, which simulates the dynamics of a 3D world. This 4D embodied world model allows us to generate realistic 3D interactions, such as grasping objects or opening drawers, with a level of detail that traditional 2D-based models cannot achieve. By modeling spatial and temporal dimensions, our model provides the depth and pose information essential for robotic manipulation. However, the task of learning a 4D embodied world model is challenging as the dynamics of the world are extremely computationally expensive to train and learn, requiring models to generate outputs in three-dimensional space and time. To efficiently represent and predict the dynamics of the world, we propose a substantially more lightweight representation of the 4D world, consisting of predicting a sequence of RGB, depth, and normal maps of the scene. This combined representation accurately captures the appearance, geometry, and surface of a scene while being substantially lower dimensional than explicitly predicting world dynamics. Furthermore, such a representation shares substantial similarities to existing video models, allowing us to directly use the generative capabilities of existing video models to effectively construct our 4D world model.

Given this intermediate representation, we present an efficient algorithm to reconstruct accurate 4D scenes from generated RGB-DN videos. For each frame, we use a combination of both depth and normal prediction to integrate a smooth 3D surface of the scene. We then use optical flow between generated frames to distinguish between background and dynamic regions in the reconstructed 3D scene across frames and introduce two novel loss functions to enforce consistency across scenes over time. As shown in Figure 1, our 4D Embodied World Model enables the construction of high-fidelity 4D-generated scenes, facilitating strong-performance action planning for downstream tasks.

A key challenge for training TesserAct is a lack of access to existing large-scale datasets with high-quality 4D

annotations, or the RGB image, depth, and normal information needed to train our approach. To overcome this, we collect a 4D embodied video dataset consisting of synthetic data with ground truth depth and normal information and real-world data with estimated depth and normal information with off-the-shelf estimators.

Overall, our paper has the following contributions:
- We collect a 4D embodied video dataset with compact and high-quality RGB, depth, and normal annotations and learn a 4D embodied world model.
- We present an algorithm to convert the generated RGB-DN video into high-quality 4D scenes coherent across both time and space.
- Extensive experiments demonstrate our 4D embodied world model can predict high-fidelity 4D scenes and achieve superior performance in downstream embodied tasks compared to traditional video-based world models.

2. Related Work

Embodied Foundation Models A flurry of recent work has focused on constructing foundation models for general purpose agents [18, 68]. One line of work has focused on constructing multimodal language models that operate over images [13, 20, 29, 39, 51, 63, 73] as well as 3D inputs [24, 25] and output text describing the actions of an agent. Other works have focused on the construction of vision-language-action (VLA) models that directly output action tokens [7, 34, 76]. Both of the previous approaches aim to construct foundation model policies (over text or continuous actions). In contrast, our work aims to instead construct a foundation 4D world model for embodied agents, which can then be used for downstream applications such as planning [14, 74] or policy synthesis [15, 40].

Learning World Models Learning dynamics model of the world given control inputs has been a long-standing challenge in system identification [42], model-based reinforcement learning [56], and optimal control [4, 81]. A large body of work focused on learning world models in the low dimensional state space [1, 17, 38], which while being efficient to learn, is difficult to generalize across many environments. Other works have explored how world models may be learned over pixel-space images [11, 12, 21, 22, 43], but such models are trained on simple game environments. With advances in generative modeling, a large flurry of recent research has focused on using video models as foundation world models [8, 46, 64, 67, 77, 79] but such models operate over the space of 2D pixels which does not fully simulate the 3D world. Most similar to our work are [76], which predicts only the 3D goal state for robotic tasks, and [57], an geometric-aware world model trained purely on synthetic data without language grounding or downstream robotic tasks; in contrast, our approach models the 4D scene from RGB-DN videos and supports language-conditioned control.

Dataset	Domain	Depth Source	Normal Source	Embodiment	# of videos
RLBench [26]	Synthetic	Simulator	Depth2Normal [2]	Franka Panda	80k
RT1 Fractal Data [6]				Google Robot	80k
Bridge [61]	Real	Rolling Depth [31]	Marigold [32]	WidowX	25k
SomethingSomethingV2 [19]				Human Hand	100k

Table 1. **Overview of the 4D embodied video datasets.**

4D Video Generation The task of 4D video generation has gained increasing attention in recent years [55, 71], driven by advancements in diffusion models [23, 47, 54], neural radiance fields [44], and 3D Gaussian splatting [33]. However, existing methods often suffer from slow optimization due to hybrid frameworks [3, 41, 65, 75] and the convergence challenges of SDS loss [30, 53]. Instead, we represent 4D scenes using RGB-DN videos, which offer more efficient training and provide high-accuracy 3D information crucial for embodied tasks. Furthermore, our approach is the first to directly predict 4D scenes from the current frame and the embodied agent's action described in the text.

3. Preliminaries

3.1. Latent Video Diffusion Models

Diffusion models [23, 54] are capable of learning the data distribution $p(x)$ by progressively adding noise to the data until it resembles a Gaussian distribution through a forward process. During inference, a denoiser ϵ is trained to recover the data from this noisy state. Latent video diffusion models [77] utilize a Variational Autoencoder (VAE) [35, 60] to encode the data in the latent space, maintaining high-quality outputs while more efficiently modeling the data distribution.

We formulate the task of RGB $\mathcal{V}$, depth $\mathcal{D}$, and normal $\mathcal{N}$ video generation as a conditional denoising generation task, i.e., we model the distribution $p(\mathbf{v}, \mathbf{d}, \mathbf{n}|\mathbf{v}^0, \mathbf{d}^0, \mathbf{n}^0, \mathcal{T})$, where $\mathbf{v}, \mathbf{d}, \mathbf{n}$ represent the predicted future latent sequences of RGB, depth, and normal maps, respectively; the conditions $\mathbf{v}^0, \mathbf{d}^0, \mathbf{n}^0, \mathcal{T}$ are the latent of RGB image, depth and normal maps, and embodied agent's action in text.

The forward diffusion process adds Gaussian noise to the latent $\mathbf{z} \in \{\mathbf{v}, \mathbf{d}, \mathbf{n}\}$ over T timesteps, defined as:

$$q(\mathbf{z}_t|\mathbf{z}_{t-1}) = \mathcal{N}\left(\mathbf{z}_t; \sqrt{\alpha_t}\mathbf{z}_{t-1}, (1-\alpha_t)\mathbf{I}\right) \quad (1)$$

where $t \in \{1, 2, \ldots, T\}$ denotes the diffusion step, α_t is a parameter controlling the noise influence at each step, and $\mathbf{I}$ is the identity matrix. In the reverse process, the model aims to recover the original latent from the noise. Let $\mathbf{x} = [\mathbf{v}, \mathbf{n}, \mathbf{d}]$ denoting the concatenation of $\mathbf{v}, \mathbf{n}, \mathbf{d}$, a denoising network $\epsilon_\theta(\mathbf{x}_t, t, \mathbf{x}^0, \mathcal{T})$ with learning parameters θ is trained to predict the noise added at each timestep. The

reverse process is defined as:

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}^0, \mathcal{T}) = \mathcal{N}\left(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t, \mathbf{x}^0, \mathcal{T}), \Sigma_\theta(\mathbf{x}_t, t)\right) \quad (2)$$

Once the denoised latent $\mathbf{x}_0$ is obtained, the model maps it back to the pixel space to obtain the final RGB-DN video.

3.2. Depth Optimization via Normal Integration

As discussed in [9, 66], normal maps provide essential information about surface orientation, which is vital for enforcing geometric constraints and imposing surface smoothness and continuity during depth optimization. This spatial optimization leads to more accurate and reliable depth estimates that closely align with the true 3D geometry and capture fine surface details.

To formalize the process, we use the perspective camera model to set constraints on the depth and surface normal. In the coordinate system of the 2D image at frame i, a pixel position is given as $\boldsymbol{u} = (u, v)^T \in \mathcal{V}^i$, and its corresponding depth scalar, normal vector is $d \in \mathcal{D}^i, \boldsymbol{n} = (n_x, n_y, n_z) \in \mathcal{N}^i$. Under the assumption of a perspective camera with focal length f and the principal point $(c_u, c_v)^T$, as proposed by [16], the log-depth $\tilde{d} = \log(d)$ should satisfy the following equations: $\tilde{n}_z \partial_u \tilde{d} + n_x = 0$ and $\tilde{n}_z \partial_v \tilde{d} + n_y = 0$ where $\tilde{n}_z = n_x(u - c_u) + n_y(v - c_v) + n_z f$. In addition, we can add the assumption that all locations are smooth surfaces [9]. We can convert the above constraint to the quadratic loss function, allowing us to find the optimized depth map:

$$\min_d \iint_\Omega (\tilde{n}_z \partial_u \tilde{d} + n_x)^2 + (\tilde{n}_z \partial_u \tilde{d} + n_y)^2 \mathrm{d}u \mathrm{d}v. \quad (3)$$

Following [9], we can convert the above objective to an iteratively optimized loss objective. At iteration step t, we can compute the matrix $W(\tilde{d}_t)$ and iteratively optimize for a refined depth prediction $\tilde{d}_{t+1}$:

$$\tilde{d}_{t+1} = \arg\min_{\tilde{d}}(A\tilde{d} - b)^T W(\tilde{d}_t)(A\tilde{d} - b) \stackrel{\text{def}}{=} \arg\min_{\tilde{\mathcal{D}}} \mathcal{L}_s(\tilde{\mathcal{D}}, \mathcal{N}^i)$$

where A and b are defined by normals and camera intrinsic.

4. Learning a 4D Embodied World Model

Learning how 3D scenes may change over time based on the current observation and action is crucial for embodied agents.
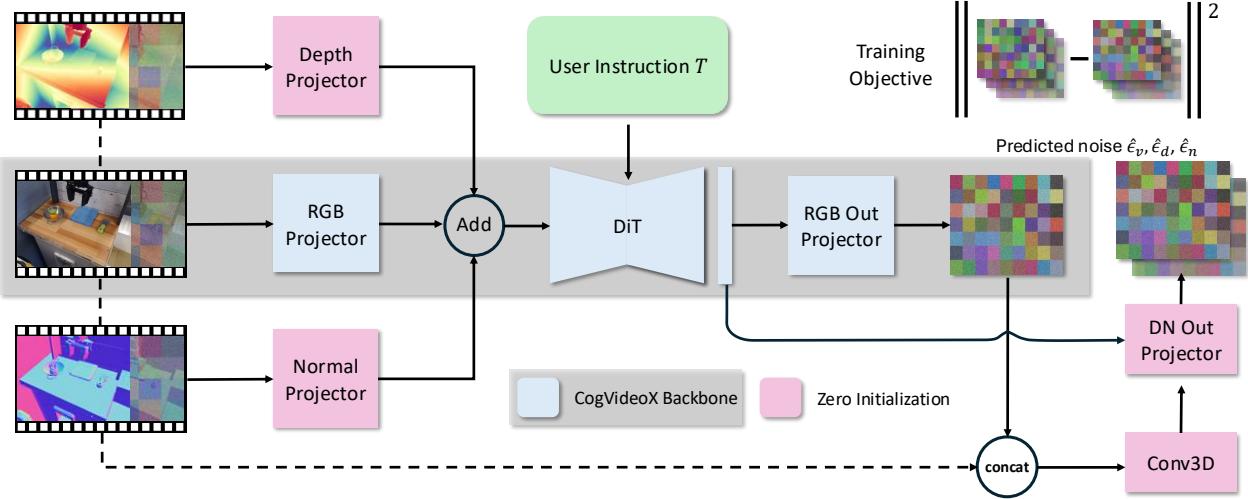
Figure 2. **Architecture and Training Overview of TesserAct.**

We propose to learn a 4D embodied world model, TesserAct, by training on RGB-DN videos and reconstructing the 4D scenes from it. We first introduce the 4D embodied video dataset we collected in Sec. 4.1, then discuss the model architecture and training strategy in Sec. 4.2. In Sec. 4.3, we propose an efficient optimization algorithm with two novel loss functions to convert the generated RGB-DN videos into 4D scenes. Finally, in Sec. 4.4, we demonstrate how the 4D world model can help downstream embodied tasks.

4.1. 4D Embodied Video Dataset

Learning 4D embodied world models requires large-scale 4D datasets, which are expensive to collect in the real world. In this section, we present a data collection and annotation pipeline that enables us to automatically construct 4D datasets from existing video datasets.

Simulator-synthesized data provide ground truth depth information, so we first selected 20 tasks of relatively high difficulty from RLBench [26] and generated 1000 instances captured from 4 different views for each task, making 80k synthetic 4D embodied videos in total. Although the simulator provides metric depth information, it lacks surface normal data. To estimate normals, we use the `depth2normal` function from DSINE [2]. To enhance the generalization, we adopt the scene randomization techniques from the Colosseum data generation pipeline [48], which alters the background, table texture and light of the scene.

While synthetic data provides depth and normal data of high quality, their diversity is limited, resulting in a gap compared to real-world scenarios. To bridge this gap, we also incorporate real-world video datasets. As most of these datasets lack depth and normal annotations, we employ the state-of-the-art video depth estimator RollingDepth [31] to annotate the videos with affine-invariant depth. As affine-invariant depth map does not directly yield normal map as metric depth does and a reliable video normal estimator is currently unavailable, we annotate normal maps using Temporal-Consistent Marigold-LCM-normal[1]. These two approaches allow us to obtain high-quality, sharp, and temporally consistent video depth and normal annotations. Specifically, we select two high-quality datasets from OpenX [58], the Fractal data [6], and the Bridge [61] dataset. Moreover, to further increase the diversity of the instructions, we incorporated the human-object interaction dataset, Something Something V2 [19]. Detailed statistics are shown in Table 1.

4.2. Model Architecture and Training Strategy

Training a diffusion model to generate temporal RGB-DN data is challenging. To effectively train RGB video models, large-scale video datasets containing billions of high-quality samples are typically employed [69, 77]. In contrast, our RGB-DN dataset, even with automatic annotation, comprises only about 200k data points, which is insufficient to train a world model from scratch. To overcome this limitation, we modify and fine-tune CogVideoX [69] to serve as our RGB-DN prediction model, leveraging the pre-trained knowledge within it to effectively bootstrap our 4D model.

Our architecture is illustrated in Figure 2. First, we utilize the 3D VAE [35, 60] from CogVideoX to separately encode the RGB ($\mathbf{v}$), depth ($\mathbf{d}$), and normal ($\mathbf{n}$) videos, without any additional fine-tuning of the VAE. These latent
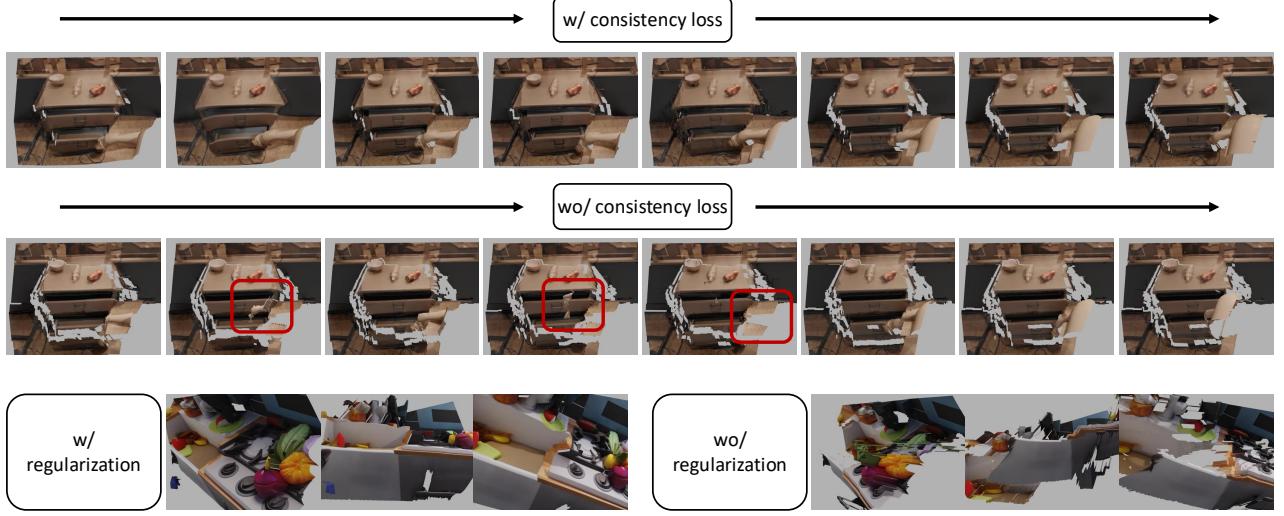

Figure 3. **Effect of consistency and regularization loss on 4D scene reconstruction.** The red boxes highlight the inconsistent regions.

representations are perturbed with noise to get $\mathbf{x}_t$, and are then fed into our model along with the corresponding image latent head $\mathbf{x}^0$. For the input design, we introduce three separate projectors for each modality to extract the embeddings: $f_{\mathbf{z}} = \texttt{InputProj}(\mathbf{z}_t, \mathbf{z}^0)$, where $\mathbf{z} \in \{\mathbf{v}, \mathbf{d}, \mathbf{n}\}$. DiT then takes the sum of these embeddings as input, conditioned on the textual input $\mathcal{T}$ and denoising step t, to obtain the hidden state: $\mathbf{h} = \texttt{DiT}(\sum f_{\mathbf{z}}, t, \mathcal{T})$. To distinguish between different robot arms, $\mathcal{T}$ is defined as [action instruction] + [robot arm name], e.g., "pick up apple *google robot*". On the output side, we retain the original RGB output method: $\epsilon_{\mathbf{v}}^* = \texttt{OutputProj}(h)$. However, we introduce additional modules for depth and normal prediction. A Conv3D layer is used to encode the concatenation of the input latents and the predicted RGB denoised output. These are then combined with the hidden states produced by the DiT backbone and passed through the output projector to obtain the denoised predictions for depth and normal: $\epsilon_{\mathbf{d},\mathbf{n}}^* = \texttt{DNProj}(h, \texttt{Conv3D}(\epsilon_{\mathbf{v}}^*, [\mathbf{z}_t; \mathbf{z}^0]_{z \in \{v,d,n\}}))$. To preserve the pretrained knowledge, we initialize our model with the CogVideoX weights. All other modules are initialized with zeros, ensuring that the RGB output at the beginning of training matches the output of CogVideoX. During training, we randomly select samples from the 4D embodied dataset $(\mathcal{V}, \mathcal{D}, \mathcal{N}, \mathcal{T})$ constructed above and apply Eq.1 to add noise $\epsilon_{\mathbf{v}}$, $\epsilon_{\mathbf{d}}$, and $\epsilon_{\mathbf{n}}$ to the RGB-DN data at timestep t, minimizing the following objective:

$$L = \mathbb{E}_{\mathbf{v}_0, \mathcal{T}, t, \epsilon} \left[\left\| [\epsilon_{\mathbf{v}}, \epsilon_{\mathbf{d}}, \epsilon_{\mathbf{n}}] - \epsilon_\theta(\mathbf{x}_t, t, \mathbf{x}^0, \mathcal{T}) \right\|^2 \right] \quad (4)$$

4.3. 4D Scene Reconstruction

After obtaining the RGB-DN video, we further optimize the depth and reconstruct the 4D scene. Similar to prior works [32, 70], our depth representation for each image is given by a relative map in the range $[0, 1]$, and thus cannot directly reconstruct the entire scene. While past work has sidestepped this by assuming either a default scale for depth [72] or by directly predicting metric depth [10, 76], such reconstructions from depth are often coarse and often cause reconstructed planes or walls to be tilted.

With the normal maps $\mathcal{N}^i$, we can optimize the depth maps $\mathcal{D}^i$ via normal integration for refined depth maps $\hat{\mathcal{D}}^i$ as introduced in Sec. 3.2 with a loss term $\mathcal{L}_s$ for spatial consistency. However, this approach optimizes depth frame by frame, which lacks temporal consistency across the dynamic scene. To address this, we compute optical flow between frames [59] $\mathcal{F} = \text{RAFT}(\mathcal{V})$ and enforce consistency of depth across frames. We define the static regions of each frame as the pixels with the magnitude of optical flow smaller than threshold c and obtain its mask by $\mathcal{M}_s^i = \|\mathcal{F}^i\| \leq c$. We then define the dynamic parts of an image as $\mathcal{M}_d^i = \neg\mathcal{M}_s^i$. We further define the background of an image as static regions that are fixed across image frames, $\mathcal{M}_b^i = \mathcal{M}_s^i \cap \mathcal{M}_s^{i-1}$

Since optical flow represents the movement of objects in the 2D-pixel space, we can retrieve the depth at any position from the previous frame to impose consistency constraints. To compute the depth values from the previous frame at positions corresponding to the current frame, we utilize the optical flow $\mathcal{F}^{i \to (i-1)}$. For each pixel (u, v) in frame i, the optical flow provides the displacement $(\Delta u, \Delta v)$, allowing us to find the corresponding pixel in frame $i-1$ at position $(u - \Delta u, v - \Delta v)$. Based on this mapping, we define the

Domain	Method	RGB			Depth			Normal			Point Cloud
		FVD ↓	SSIM ↑	PSNR ↑	AbsRel ↓	δ_1 ↑	δ_2 ↑	Mean ↓	Median ↓	11.25° ↑	Chamfer L_1 ↓
Real	4D Point-E	-	-	-	-	-	-	-	-	-	0.2211
	OpenSora	23.67	71.31	19.25	31.41	60.39	79.97	41.82	32.15	13.58	0.3013
	CogVideoX	**20.64**	**79.38**	**22.39**	<u>26.17</u>	<u>64.82</u>	<u>81.62</u>	<u>19.53</u>	<u>10.09</u>	<u>22.70</u>	<u>0.2191</u>
	TesserAct (Ours)	<u>21.59</u>	<u>75.86</u>	<u>20.27</u>	**22.07**	**66.80**	**82.60**	**15.74**	**7.32**	**27.80**	**0.2030**
Synthetic	4D Point-E	-	-	-	-	-	-	-	-	-	<u>0.1086</u>
	OpenSora	54.11	65.90	19.28	<u>18.40</u>	<u>65.02</u>	<u>91.20</u>	**12.94**	<u>7.58</u>	25.02	0.2570
	CogVideoX	<u>41.23</u>	<u>76.60</u>	**20.87**	19.81	60.07	80.16	20.36	10.47	<u>26.04</u>	0.2884
	TesserAct (Ours)	**40.01**	**77.59**	<u>19.73</u>	**16.02**	**69.26**	**93.03**	<u>14.75</u>	**6.34**	**36.85**	**0.0811**

Table 2. **Main results on the 4D scene generation.** All metrics are averaged over 10 runs for each of the samples. The best results are in **bold**, and the second best are in <u>underlined</u>. TesserAct predicts the depth and normal maps most accurately without hurting RGB much and thus achieves the best accuracy of reconstructed 4D point clouds across real and synthetic image domains.

$\mathcal{D}^{i\to(i-1)}$ such that: $\mathcal{D}^{i\to(i-1)}(u,v) = \mathcal{D}^{i-1}(u - \Delta u, v - \Delta v)$. We then introduce the consistency loss over the dynamic and background region of the image $\mathcal{L}_c$:

$$\mathcal{L}_c(\tilde{\mathcal{D}}, \hat{\mathcal{D}}^{i-1}, \mathcal{F}^i, \mathcal{F}^{i-1}) = \lambda_{cd} \left\| \tilde{\mathcal{D}}^i \circ \mathcal{M}_d^i - \mathcal{D}^{i\to(i-1)} \circ \mathcal{M}_d^i \right\|^2 + \\ \lambda_{cb} \left\| \tilde{\mathcal{D}}^i \circ \mathcal{M}_b^i - \mathcal{D}^{i\to(i-1)} \circ \mathcal{M}_b^i \right\|^2 \quad (5)$$

In addition to the consistency loss, we also incorporate the regularization loss $\mathcal{L}_r$, enforcing that optimized depths are similar to the generated depth map $\mathcal{D}^i$:

$$\mathcal{L}_r(\tilde{\mathcal{D}}, \mathcal{D}^i) = \lambda_{rd} \left\| \tilde{\mathcal{D}}^i \circ \mathcal{M}_d^i - \mathcal{D}^i \circ \mathcal{M}_d^i \right\|^2 + \lambda_{rb} \left\| \tilde{\mathcal{D}}^i \circ \mathcal{M}_b^i - \mathcal{D}^i \circ \mathcal{M}_b^i \right\|^2 \quad (6)$$

The overall loss objective we optimize is given by:

$$\arg\min_{\tilde{\mathcal{D}}} \quad \mathcal{L}_s(\tilde{\mathcal{D}}, \mathcal{N}^i) + \mathcal{L}_c(\tilde{\mathcal{D}}, \hat{\mathcal{D}}^{i-1}, \mathcal{F}^i, \mathcal{F}^{i-1}) + \mathcal{L}_r(\tilde{\mathcal{D}}, \mathcal{D}^i) \quad (7)$$

Starting from the first frame, we iteratively refine the depth map by optimizing the loss above, and initialize the depth map at frame i with the generated depth map $\tilde{D}_0 = \mathcal{D}^i$. With refined depth maps $\tilde{\mathcal{D}}$ and RGB Images $\mathcal{V}$, we can reconstruct 4D point clouds representing the world that are consistent over both space and time.

4.4. Embodied Action Planning with 4D Scenes

After generating 4D scenes, which encapsulate both spatial and temporal information, we extract geometric details that can significantly enhance downstream tasks in robotics. The detailed geometry captured by these scenes plays a crucial role in tasks including robotic grasping.

To achieve this, we employ an inverse dynamics model built on the 4D point clouds, predicting the appropriate robot action a_i based on the current state s_i, the predicted future state s_{i+1} and the instruction $\mathcal{T}$. Mathematically, this relationship is expressed as $a_i = \text{ID}(s_i, s_{i+1}, \mathcal{T})$, where s_i represents the scene at the time step i. Specifically, we use PointNet [49] to encode the point cloud and extract 3D features. These features are then combined with the instruction text embeddings and further processed by an MLP to generate the final 7-DoF action.

5. Experiments

We first evaluate the quality of 4D scene predictions from our model across real and synthetic datasets in Sec. 5.1, then conduct experiments in RLBench to demonstrate how the 4D information helps the embodied tasks in Sec. 5.2. We provide more qualitative results and videos in the Supplementary and the website.

5.1. 4D Scene Prediction

5.1.1. Setup

Datasets. We conduct experiments on the real domain with 400 unseen samples in RT1 Fractal [6] and Bridge [61] dataset where depth and normal are estimated as in Sec. 4.1, and the synthetic domain with 200 unseen samples in RL-Bench [26], where ground truth depth and normal maps are directly accessible.

Metrics. We evaluate the video quality with FVD, SSIM, and PSNR; depth quality with AbsRel, δ_1, and δ_2; normal maps quality with Mean, Median, and 11.25°; reconstructed point cloud quality with the L_1 Chamfer Distance. We generate 10 times per sample and report the average.

Baselines. We compare our method to two video diffusion models and a 4D point cloud diffusion model.

- OpenSora [77], video diffusion model fine-tuned with LoRA on the same dataset without depth and normal annotations. To construct the full 4D scene, depth and normal are additional estimated given the predicted video with Rolling Depth and Marigold.
- CogVideoX [69], video diffusion model fine-tuned with LoRA on the same dataset without depth and normal annotations. The 4D scene is obtained similarly to the above.
- 4D Point-E, since no prior work directly generates dynamic scenes from the first frame and text inputs, we implemented a 4D point cloud diffusion model as the baseline, where we modify the Point-E [45] model by conditioning it on the mean of CLIP [50] features extracted from both text and image inputs, outputting T point clouds

Methods	close box	open drawer	open jar	open microwave	put knife	sweep to dustpan	lid off	weighing off	water plants
Image-BC	53	4	0	5	0	0	12	21	0
UniPi*	81	67	38	**72**	66	49	70	**68**	35
4DWM (Ours)	**88**	**80**	**44**	70	**70**	**56**	**73**	62	**41**

Table 3. **TesserAct boosts the performance of action planning.** We report the success rate averaged over 100 episodes for each task here.

Method	PSNR	SSIM	LPIPS	CLIP Score	CLIP Aesthetic	Time Costs
SoM[62]	10.94	24.02	**73.82**	66.67	3.61	∼2 hours
Ours	**12.99**	**42.62**	60.51	**83.02**	**3.73**	**∼ 1 mins**

Table 4. **Novel view synthesis results on RLBench.**

of size n, where T is set to 4 and n is set to 8192 due to computational constraints.

Implementation Details. We train our model on the collected 4D embodied video dataset using a multi-resolution training approach, predicting 49 frames at a time. For more details, please kindly refer to the Supplementary Materials.

5.1.2. Results and Analysis

TesserAct predicts high quality 4D scenes. As shown in Table 2, our model accurately predicts the depth and normal maps compared to video diffusion models with post-estimation, verifying the effectiveness of our learned model. With better depth and normal maps, the 4D point clouds reconstructed with our method achieve the lowest Chamfer distances across real and synthetic datasets. The 4D Point-E method performs better than video diffusion models, particularly on RLBench, but still lags behind our approach. Additionally, training directly with point clouds is computationally expensive, restricting the number of frames used. In contrast, our model leverages an efficient representation with RGB-DN videos to generate more precise 4D scenes, particularly in capturing fine-grained details in dynamic scenes. We also show qualitative results in Figure 4 (a).

TesserAct synthesizes novel views efficiently. Our method could also perform monocular video to 4D tasks by predicting depth and normal sequences and generating point clouds. We conduct experiments on the task of novel view synthesis on RLBench and compare with Shape of Motion [62], a state-of-the-art video reconstruction approach that utilizes Gaussian splatting [33]. The input is a monocular front camera video, and we compare results from the overhead and left shoulder cameras. We report PSNR (reconstruction accuracy), SSIM (structural similarity), LPIPS (perceptual difference), CLIP Score (semantic match) [78], CLIP aesthetic (visual quality) [37], and Time costs in Table 4. The results show our method can synthesize novel views of higher visual quality and better alignment in signif-

icantly less time. A qualitative result on the Bridge dataset is shown in Figure 4 (c).

Consistency and Regularization Loss are effective. We conduct ablation experiments on our newly designed loss terms in Sec. 4.3. The results are shown in Figure 3. The first two rows demonstrate the effect of the consistency loss, where we render frames from the same camera view at different time steps. The results show that the robot arm's movements are more coherent with the consistency loss applied. The last row highlights the role of the regularization loss. We display images of the same frame from three different views, revealing that this loss term helps improve the geometric accuracy of the reconstruction.

TesserAct shows generalization across scenes and embodiments. Our model demonstrates strong generalization capabilities. Benefiting from the knowledge of CogVideoX, the model achieves good generation on unseen scenes and unseen objects. Additionally, it performs well in cross-embodiment scenarios, such as using the Bridge dataset robotic arm in the RT-1 environment. Figure 4 (b) shows a generalization result on unseen scenes and objects. More results can be found in the Supplementary Materials.

5.2. Embodied Action Planning

Dataset. We select 9 challenging tasks from RLBench [26] including tasks requiring high-precision grasping.

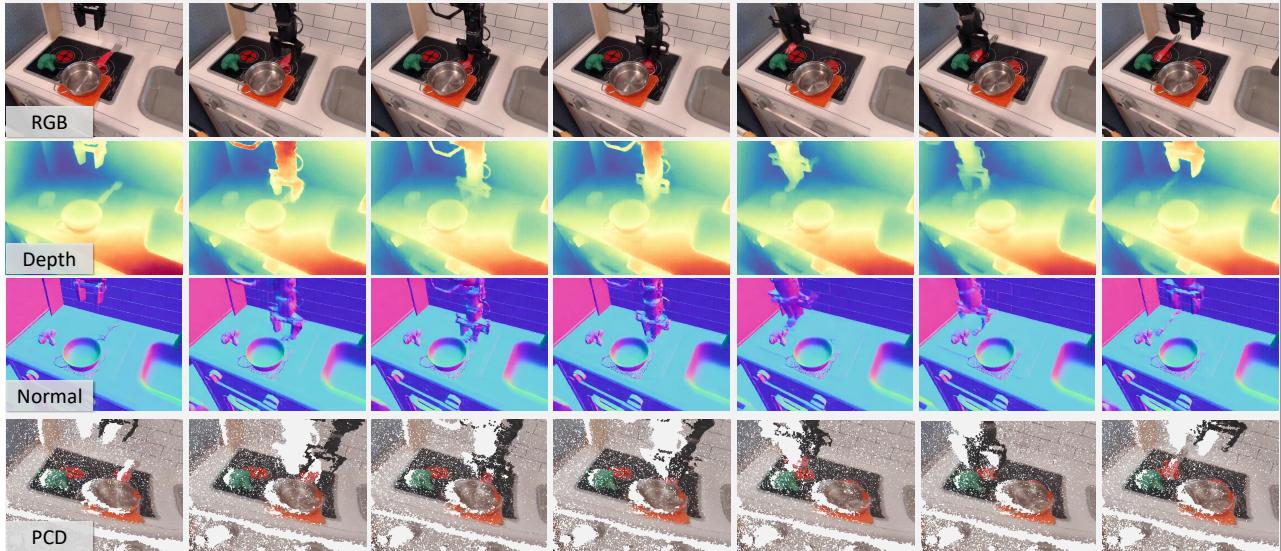**Metric.** The success rate averaged over 100 episodes.

Baseline. We compare our method to a behavior cloning agent and a video-based world model.

- Image-BC [27]: a behavior cloning agent that takes in an image and task instruction and outputs the 7-DoF actions.
- UniPi* [15]: a method that takes the task instruction and current image, predicts the future video, and uses a 2D-based inverse dynamic policy to predict actions. For this baseline, we re-implement it by replacing the backbone with fine-tuned CogVideoX [69] for fair comparison.

Implementation Details. We collected 500 samples for each task to train the inverse dynamic model. Given an initial state during inference, we first predict and record all future keyframes. In subsequent actions, we only query the inverse dynamic model to obtain the corresponding actions by the current state and the predicted future state. We post-trained the model from Sec. 5.1, allowing the model to predict only the keyframes for each task in RLBench. Our maximum
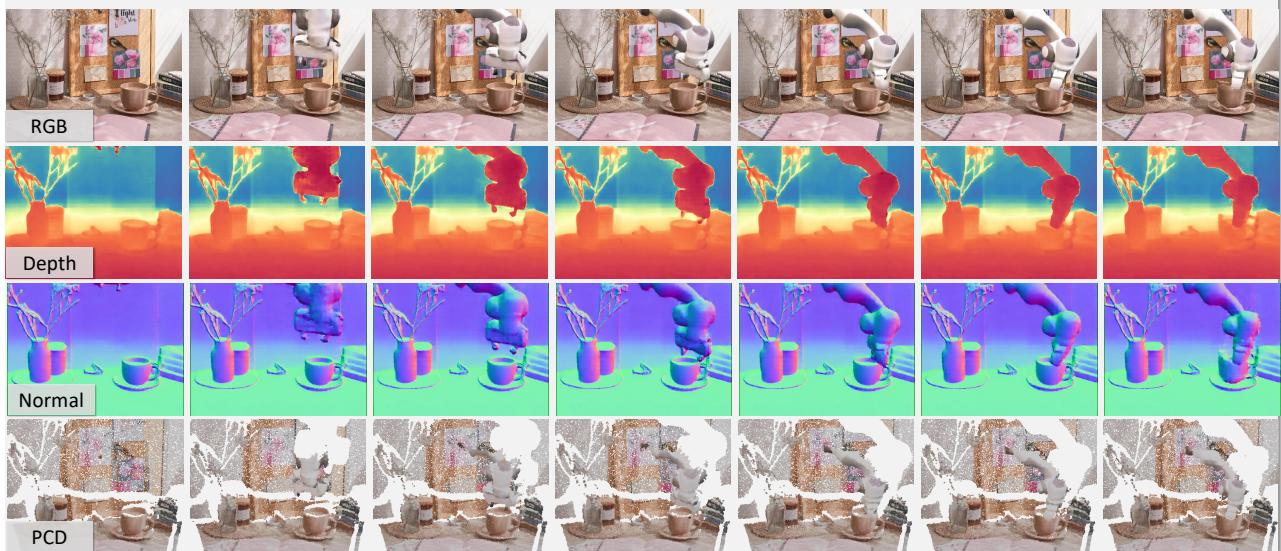
(a) In-domain 4D Generation Results

Input instruction: Place the red fork to the left of the left burner – *Trossen WidowX 250*

RGB

Depth

Normal

PCD

(b) Out-of-domain 4D Generation Results

Input instruction: Pick up brown cup – *Franka Panda Robot*

RGB

Depth

Normal

PCD

(c) Novel View Synthesis

Input View

Novel View

Figure 4. **Qualitative results of** (a) **In-domain 4D generation results.** (b) **Generalization over unseen scenes and objects.** (c) **Novel view synthesis.**

frame length is 13, with a fixed resolution of 512×512.

5.2.1. Results and Analysis

The results are shown in Table 3, where our method outperforms video diffusion models and image behavior cloning agents in most of the tasks. This is because, in most tasks, 4D point clouds can reveal the geometry of objects, providing better spatial guidance for robotics planning, as seen in tasks like `close box` and `open jar`. At the same time, 3D information can assist with tool use, such as in tasks like `sweep to dustpan` and `water plants`. However, in the `open microwave` and `weighing off` tasks, the performance is not as good as the baseline, possibly because these tasks already have sufficient information in the 2D front image. Overall, these results highlight the potential of combining 4D scene prediction with inverse dynamic models to improve robotics task execution.

6. Conclusion

We learn a 4D generative world model, TesserAct, using a collected 4D embodied video dataset, which consists of robotic manipulation videos annotated with depth and normal information. To ensure both temporal and spatial consistency in scene reconstruction, we introduce two novel loss terms. Our experiments across synthetic and real-world datasets demonstrate that our model generates high-quality 4D scenes and significantly enhances the performance of downstream embodied tasks by leveraging 3D information. We believe that such world models will become increasingly powerful and essential, serving as a foundation for simulating the physical world and advancing the development of intelligent embodied agents. These models will enable fully offline policy training in the real world and facilitate planning through imagined rollouts within the learned world representation.

7. Limitations

While our RGB-DN representation of a 4D world model is cheap and easy to predict, it only captures a single surface of the world. To construct a more complete 4D world model, it may be interesting in the future to have a generative model that generates multiple RGB-DN views of the world, which can then be integrated to form a more complete 4D world model.

Acknowledgements

References

[1] Alessandro Achille and Stefano Soatto. A separation principle for control in the age of deep learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 1:287–307, 2018. 2

[2] Gwangbin Bae and Andrew J. Davison. Rethinking inductive biases for surface normal estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 3, 4

[3] Sherwin Bahmani, Ivan Skorokhodov, Victor Rong, Gordon Wetzstein, Leonidas Guibas, Peter Wonka, Sergey Tulyakov, Jeong Joon Park, Andrea Tagliasacchi, and David B Lindell. 4d-fy: Text-to-4d generation using hybrid score distillation sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7996–8006, 2024. 3

[4] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 1995. 2

[5] Shariq Farooq Bhat, Reiner Birkl, Diana Wofk, Peter Wonka, and Matthias Müller. Zoedepth: Zero-shot transfer by combining relative and metric depth. *arXiv preprint arXiv:2302.12288*, 2023. 14

[6] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022. 3, 4, 6, 13, 18

[7] 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. 2

[8] 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. 2

[9] Xu Cao, Hiroaki Santo, Boxin Shi, Fumio Okura, and Yasuyuki Matsushita. Bilateral normal integration. In *European Conference on Computer Vision*, pages 552–567. Springer, 2022. 3

[10] Boyuan Chen, Zhuo Xu, Sean Kirmani, Brain Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14455–14465, 2024. 5

[11] Chang Chen, Yi-Fu Wu, Jaesik Yoon, and Sungjin Ahn. Transdreamer: Reinforcement learning with transformer world models. *arXiv preprint arXiv:2202.09481*, 2022. 2

[12] Silvia Chiappa, Sébastien Racaniere, Daan Wierstra, and Shakir Mohamed. Recurrent environment simulators. *arXiv preprint arXiv:1704.02254*, 2017. 2

[13] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: An

embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023. 2

[14] 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. 2

[15] 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, 2024. 2, 7

[16] Jean-Denis Durou, Jean-François Aujol, and Frédéric Courteille. Integrating the normal field of a surface in the presence of discontinuities. In *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 261–273. Springer, 2009. 3

[17] Norm Ferns, Prakash Panangaden, and Doina Precup. Metrics for finite markov decision processes. In *UAI*, pages 162–169, 2004. 2

[18] Roya Firoozi, Johnathan Tucker, Stephen Tian, Anirudha Majumdar, Jiankai Sun, Weiyu Liu, Yuke Zhu, Shuran Song, Ashish Kapoor, Karol Hausman, et al. Foundation models in robotics: Applications, challenges, and the future. *The International Journal of Robotics Research*, page 02783649241281508, 2023. 2

[19] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *Proceedings of the IEEE international conference on computer vision*, pages 5842–5850, 2017. 3, 4

[20] Maitrey Gramopadhye and Daniel Szafir. Generating executable action plans with environmentally-aware language models. *arXiv preprint arXiv:2210.04964*, 2022. 2

[21] David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2018. 2

[22] Danijar Hafner, Timothy P Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. In *International Conference on Learning Representations*, 2021. 2

[23] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 3

[24] Yining Hong, Haoyu Zhen, Peihao Chen, Shuhong Zheng, Yilun Du, Zhenfang Chen, and Chuang Gan. 3d-llm: Injecting the 3d world into large language models. *Advances in Neural Information Processing Systems*, 36:20482–20494, 2023. 2

[25] Jiangyong Huang, Silong Yong, Xiaojian Ma, Xiongkun Linghu, Puhao Li, Yan Wang, Qing Li, Song-Chun Zhu, Baoxiong Jia, and Siyuan Huang. An embodied generalist agent in 3d world. *arXiv preprint arXiv:2311.12871*, 2023. 2

[26] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J. Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 2020. 3, 4, 6, 7, 13

[27] Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pages 991–1002. PMLR, 2022. 7

[28] Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022. 2

[29] Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. Vima: General robot manipulation with multimodal prompts. *arXiv preprint arXiv:2210.03094*, 2(3):6, 2022. 2

[30] Yanqin Jiang, Li Zhang, Jin Gao, Weimin Hu, and Yao Yao. Consistent4d: Consistent 360 {\deg} dynamic object generation from monocular video. *arXiv preprint arXiv:2311.02848*, 2023. 3

[31] Bingxin Ke, Dominik Narnhofer, Shengyu Huang, Lei Ke, Torben Peters, Katerina Fragkiadaki, Anton Obukhov, and Konrad Schindler. Video depth without video models, 2024. 3, 4

[32] Bingxin Ke, Anton Obukhov, Shengyu Huang, Nando Metzger, Rodrigo Caye Daudt, and Konrad Schindler. Repurposing diffusion-based image generators for monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9492–9502, 2024. 3, 5

[33] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 3, 7

[34] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024. 2

[35] Diederik P Kingma. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 3, 4

[36] Frank Klinker. Exponential moving average versus moving exponential average. *Mathematische Semesterberichte*, 58: 97–107, 2011. 13

[37] LAION-AI. Aesthetic predictor. https://github.com/LAION-AI/aesthetic-predictor, 2022. 7

[38] Timothée Lesort, Natalia Díaz-Rodríguez, Jean-Franois Goudou, and David Filliat. State representation learning for control: An overview. *Neural Networks*, 108:379–392, 2018. 2

[39] Shuang Li, Xavier Puig, Chris Paxton, Yilun Du, Clinton Wang, Linxi Fan, Tao Chen, De-An Huang, Ekin Akyürek, Anima Anandkumar, et al. Pre-trained language models for interactive decision-making. *Advances in Neural Information Processing Systems*, 35:31199–31212, 2022. 2

[40] Junbang Liang, Ruoshi Liu, Ege Ozguroglu, Sruthi Sudhakar, Achal Dave, Pavel Tokmakov, Shuran Song, and Carl Vondrick. Dreamitate: Real-world visuomotor policy learning via video generation. *arXiv preprint arXiv:2406.16862*, 2024. 2

[41] Huan Ling, Seung Wook Kim, Antonio Torralba, Sanja Fidler, and Karsten Kreis. Align your gaussians: Text-to-4d

with dynamic 3d gaussians and composed diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8576–8588, 2024. 3

[42] Lennart Ljung and Torkel Glad. *Modeling of dynamic systems*. Prentice-Hall, Inc., 1994. 2

[43] Vincent Micheli, Eloi Alonso, and François Fleuret. Transformers are sample efficient world models. *arXiv preprint arXiv:2209.00588*, 2022. 2

[44] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 3

[45] Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, and Mark Chen. Point-e: A system for generating 3d point clouds from complex prompts. *arXiv preprint arXiv:2212.08751*, 2022. 6

[46] OpenAI. Video generation models as world simulators. https://openai.com/index/video-generation-models-as-world-simulators/, 2023. Accessed: 2024-10-01. 2

[47] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023. 3

[48] Wilbert Pumacay, Ishika Singh, Jiafei Duan, Ranjay Krishna, Jesse Thomason, and Dieter Fox. The colosseum: A benchmark for evaluating generalization for robotic manipulation. 2024. 4

[49] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017. 6

[50] 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. 6

[51] Shreyas Sundara Raman, Vanya Cohen, Eric Rosen, Ifrah Idrees, David Paulius, and Stefanie Tellex. Planning with large language models via corrective re-prompting. *arXiv preprint arXiv:2211.09935*, 2022. 2

[52] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International conference on machine learning*, pages 8821–8831. Pmlr, 2021. 14

[53] Jiawei Ren, Liang Pan, Jiaxiang Tang, Chi Zhang, Ang Cao, Gang Zeng, and Ziwei Liu. Dreamgaussian4d: Generative 4d gaussian splatting. *arXiv preprint arXiv:2312.17142*, 2023. 3

[54] 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. 3

[55] Uriel Singer, Shelly Sheynin, Adam Polyak, Oron Ashual, Iurii Makarov, Filippos Kokkinos, Naman Goyal, Andrea Vedaldi, Devi Parikh, Justin Johnson, et al. Text-to-4d dynamic scene generation. *arXiv preprint arXiv:2301.11280*, 2023. 3

[56] Richard S Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *ACM Sigart Bulletin*, 2(4):160–163, 1991. 2

[57] Aether Team, Haoyi Zhu, Yifan Wang, Jianjun Zhou, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Chunhua Shen, Jiangmiao Pang, et al. Aether: Geometric-aware unified world modeling. *arXiv preprint arXiv:2503.18945*, 2025. 2

[58] Open X-Embodiment Team. Open X-Embodiment: Robotic learning datasets and RT-X models. https://arxiv.org/abs/2310.08864, 2023. 4

[59] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 402–419. Springer, 2020. 5

[60] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. 3, 4

[61] Homer Walke, Kevin Black, Abraham Lee, Moo Jin Kim, Max Du, Chongyi Zheng, Tony Zhao, Philippe Hansen-Estruch, Quan Vuong, Andre He, Vivek Myers, Kuan Fang, Chelsea Finn, and Sergey Levine. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning (CoRL)*, 2023. 3, 4, 6, 13, 18

[62] Qianqian Wang, Vickie Ye, Hang Gao, Jake Austin, Zhengqi Li, and Angjoo Kanazawa. Shape of motion: 4d reconstruction from a single video. *arXiv preprint arXiv:2407.13764*, 2024. 7

[63] Zihao Wang, Shaofei Cai, Guanzhou Chen, Anji Liu, Xiaojian Ma, and Yitao Liang. Describe, explain, plan and select: interactive planning with llms enables open-world multi-task agents. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 2

[64] Jiannan Xiang, Guangyi Liu, Yi Gu, Qiyue Gao, Yuting Ning, Yuheng Zha, Zeyu Feng, Tianhua Tao, Shibo Hao, Yemin Shi, et al. Pandora: Towards general world model with natural language actions and video states. *arXiv preprint arXiv:2406.09455*, 2024. 2

[65] Yiming Xie, Chun-Han Yao, Vikram Voleti, Huaizu Jiang, and Varun Jampani. Sv4d: Dynamic 3d content generation with multi-frame and multi-view consistency. *arXiv preprint arXiv:2407.17470*, 2024. 3

[66] Yuliang Xiu, Jinlong Yang, Xu Cao, Dimitrios Tzionas, and Michael J. Black. ECON: Explicit Clothed humans Optimized via Normal integration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 3

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

[68] Sherry Yang, Ofir Nachum, Yilun Du, Jason Wei, Pieter Abbeel, and Dale Schuurmans. Foundation models for decision making: Problems, methods, and opportunities. *arXiv preprint arXiv:2303.04129*, 2023. 2

[69] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024. 4, 6, 7, 13

[70] Chongjie Ye, Lingteng Qiu, Xiaodong Gu, Qi Zuo, Yushuang Wu, Zilong Dong, Liefeng Bo, Yuliang Xiu, and Xiaoguang Han. Stablenormal: Reducing diffusion variance for stable and sharp normal. *arXiv preprint arXiv:2406.16864*, 2024. 5

[71] Yuyang Yin, Dejia Xu, Zhangyang Wang, Yao Zhao, and Yunchao Wei. 4dgen: Grounded 4d content generation with spatial-temporal consistency. *arXiv preprint arXiv:2312.17225*, 2023. 3

[72] Hong-Xing Yu, Haoyi Duan, Charles Herrmann, William T. Freeman, and Jiajun Wu. Wonderworld: Interactive 3d scene generation from a single image. *arXiv:2406.09394*, 2024. 5

[73] Hongxin Zhang, Weihua Du, Jiaming Shan, Qinhong Zhou, Yilun Du, Joshua B Tenenbaum, Tianmin Shu, and Chuang Gan. Building cooperative embodied agents modularly with large language models. *arXiv preprint arXiv:2307.02485*, 2023. 2

[74] Hongxin Zhang, Zeyuan Wang, Qiushi Lyu, Zheyuan Zhang, Sunli Chen, Tianmin Shu, Yilun Du, and Chuang Gan. Combo: Compositional world models for embodied multi-agent cooperation. *arXiv preprint arXiv:2404.10775*, 2024. 2

[75] Yuyang Zhao, Zhiwen Yan, Enze Xie, Lanqing Hong, Zhenguo Li, and Gim Hee Lee. Animate124: Animating one image to 4d dynamic scene. *arXiv preprint arXiv:2311.14603*, 2023. 3

[76] Haoyu Zhen, Xiaowen Qiu, Peihao Chen, Jincheng Yang, Xin Yan, Yilun Du, Yining Hong, and Chuang Gan. 3d-vla: A 3d vision-language-action generative world model. *arXiv preprint arXiv:2403.09631*, 2024. 2, 5, 14

[77] Zangwei Zheng, Xiangyu Peng, Tianji Yang, Chenhui Shen, Shenggui Li, Hongxin Liu, Yukun Zhou, Tianyi Li, and Yang You. Open-sora: Democratizing efficient video production for all, 2024. 2, 3, 4, 6

[78] SUN Zhengwentai. clip-score: CLIP Score for PyTorch. https://github.com/taited/clip-score, 2023. Version 0.1.1. 7

[79] Siyuan Zhou, Yilun Du, Jiaben Chen, Yandong Li, Dit-Yan Yeung, and Chuang Gan. Robodreamer: Learning compositional world models for robot imagination. *arXiv preprint arXiv:2404.12377*, 2024. 2

[80] 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. 2

[81] Karl J. Åström and Björn Wittenmark. Adaptive control of linear time-invariant systems. *Automatica*, 9(6):551–564, 1973. 2

TesserAct: Learning 4D Embodied World Models

Supplementary Material

1. Implementation Details

1.1. Video Diffusion Model Details

We trained an RGB-DN video diffusion model using the CogVideoX [69] architecture. On the input side, our depth normal projector and RGB projector shared the same architecture. On the output side, our `Conv3DNet` consisted of three layers, while the MLP had two layers, both with a dimension of 1024. The model outputs videos with 49 frames, utilizing gradient checkpointing to optimize memory usage. We set a global batch size of 16 and used `bf16` precision to accelerate. For sampling, we employed the DDPM scheduler across 50 steps and set a classifier-free guidance scale of 7.5.

The training spanned 40,000 iterations with an initial learning rate of 1×10^{-4}, a gradient clipping of 1.0, and a 1,000-step warmup. The optimizer used was Adam with ϵ set to 1×10^{-15}, and an exponential moving average (EMA [36]) decay of 0.99 was applied to stabilize training.

1.2. 4D Scene Generation

The parameters for the loss term in Eq.12 are set differently for the RT-1 [6], Bridge [61] and RLBench [26] datasets, as shown in the table below. It is worth noting that the selection of λ varies for different scenarios. In practice, achieving the best performance requires tuning these parameters.

Dataset	λ_d	λ_b	λ_{g1}	λ_{g2}
RT-1, Bridge	20	200	20	20
RLBench	20	200	2	2

Table 5. Loss Term Parameters for RT-1 and RLBench Datasets

In Figure 5, we present a visualization of the 3D robotic scene reconstruction optimized using our proposed method in the BridgeV2 [61] dataset. After estimating the depth and normal with the estimator, we refine the outputs to reconstruct the scene accurately. The figure includes untextured rendering and texture-rendered views, where the wall textures are significantly enhanced due to normal optimization. The side perspective view shows the improved shape and geometry reconstruction. Notably, the wall and table surfaces are well-aligned, appearing perpendicular to each other, further validating the effectiveness of our optimization process in capturing accurate spatial relationships.
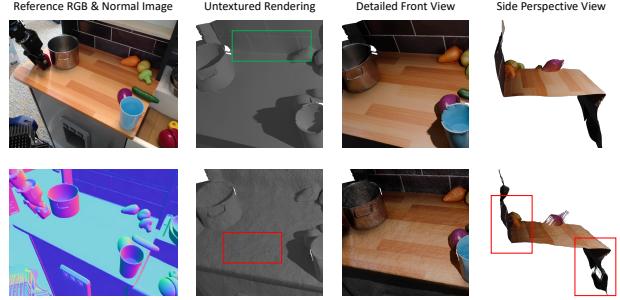

Figure 5. Visualization of the optimized 3D robotic scene reconstruction using our method. The untextured renderings show enhanced detail (green box) and improved surface smoothness (red box). The side perspective view highlights accurate shape and geometry optimization, including the perpendicular alignment of the wall and table (red boxes).

1.3. Implementation Details for Robotics Planning

For the RLBench training, we adopted the same architecture and methods as our video diffusion model, with the primary difference being that we used 13 frames and fine-tuned the model.

For the action prediction stage, we first filter out the background and floor from the data, focusing only on the points of the table and the objects manipulated by the robotic arm, and then sample 8192 points from the filtered point cloud. In our inverse dynamic model, the PointNet extracts features from this point cloud, concatenated with the instruction's language embedding, and passed into a 4-layer MLP, finally outputting the 7DoF actions. To augment the data and better adapt to the output of video diffusion models, we add significant Gaussian noise (with a relative magnitude of 20%) to both the image and point cloud coordinates.
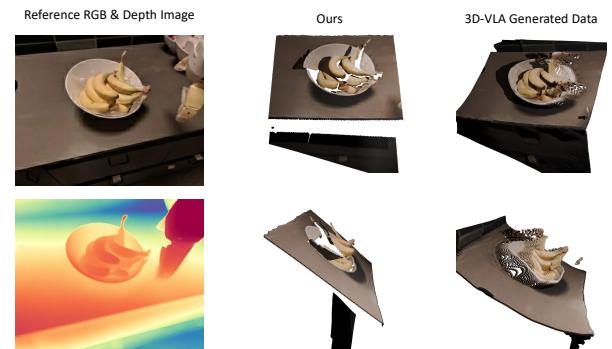

Figure 6. Comparison of point cloud generation quality between our method and 3D-VLA
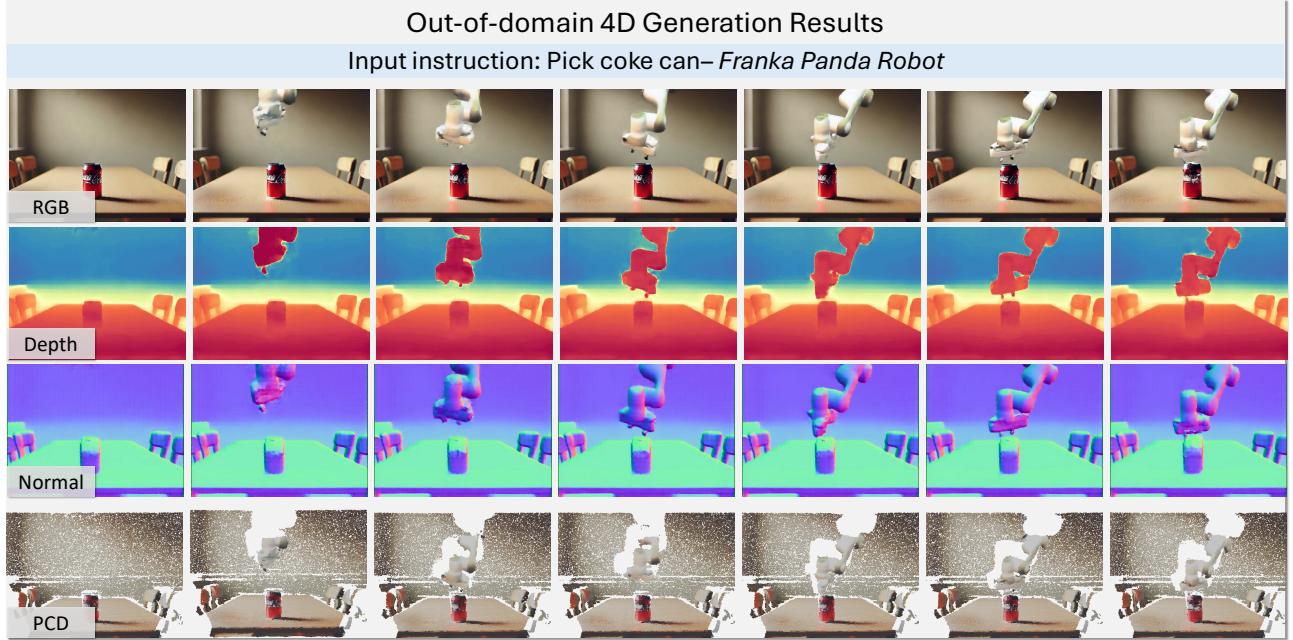
Figure 7. Out-of-domain 4D generation results

2. More Qualitative Results

2.1. Data Annotation

In this section, we first compare our data generation method with 3D-VLA [76]. They use ZoeDepth [5] for depth map estimation and directly map them into 3D space. The comparison results, shown in Figure 6, evaluate the quality of point cloud generation for both methods, with cubes replacing vertices for rendering. Our generated data demonstrates higher realism, while 3D-VLA exhibits noticeable shape distortion. Figure 12 showcases some of the RGB, depth, and normal images from the datasets we used, along with the corresponding natural language instructions.

2.2. 4D Generation

As shown in Figure 7, we present our out-of-domain results. We used DALL-E [52] to generate an image and prompted the 4D world model for generation. Our video diffusion model directly produces the RGB, depth, and normal maps, while the point clouds are rendered from the reconstructed outputs. These results highlight the robustness of our method across different visual modalities. *Additional video results* can be found in the supplementary materials folder for further analysis and evaluation.

2.3. Video Generation

We present the video generation results on the RT1, Bridge and RLBench datasets in Figure 11, Figure 9 and Figure 10. More videos can be found in our supplementary folder.

2.4. Explicit Action Planning

One potential application of our generated mesh is to extract action trajectories directly. As illustrated in Figure 8, we track the robotic arm in the video to capture its motion path. This trajectory is subsequently lifted into 3D space, enabling the reconstruction of the robot arm's action trajectory. The red line in the visualization represents the captured action trajectory.

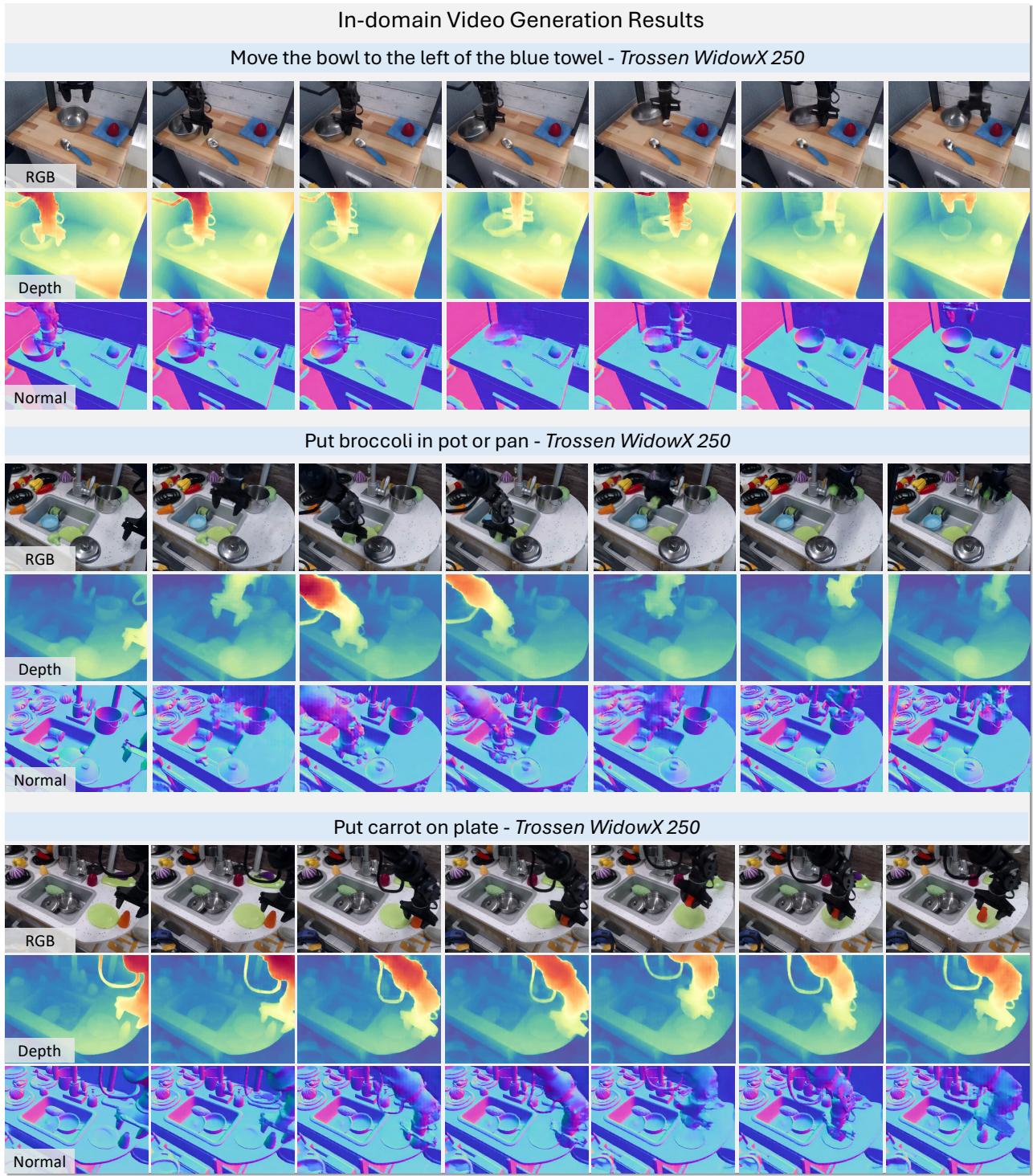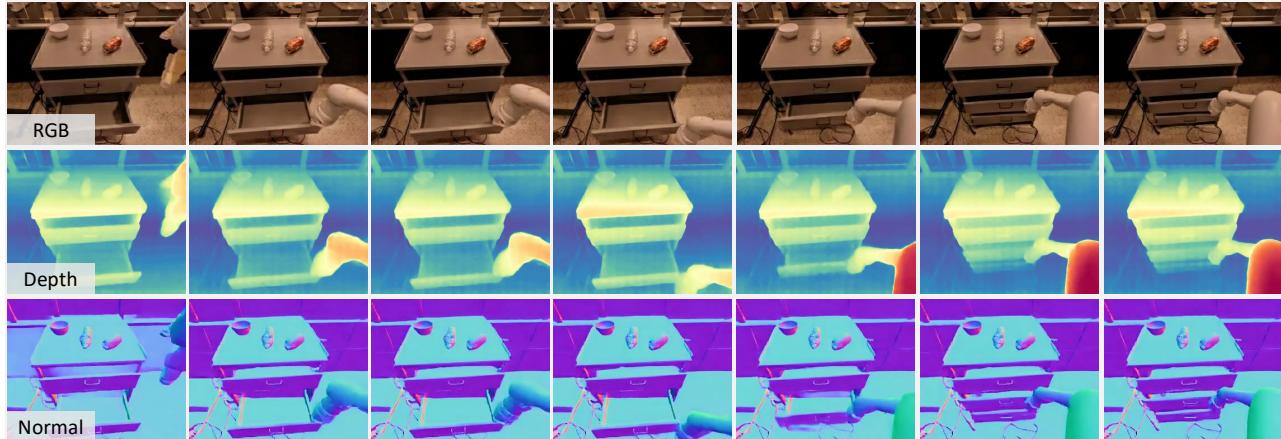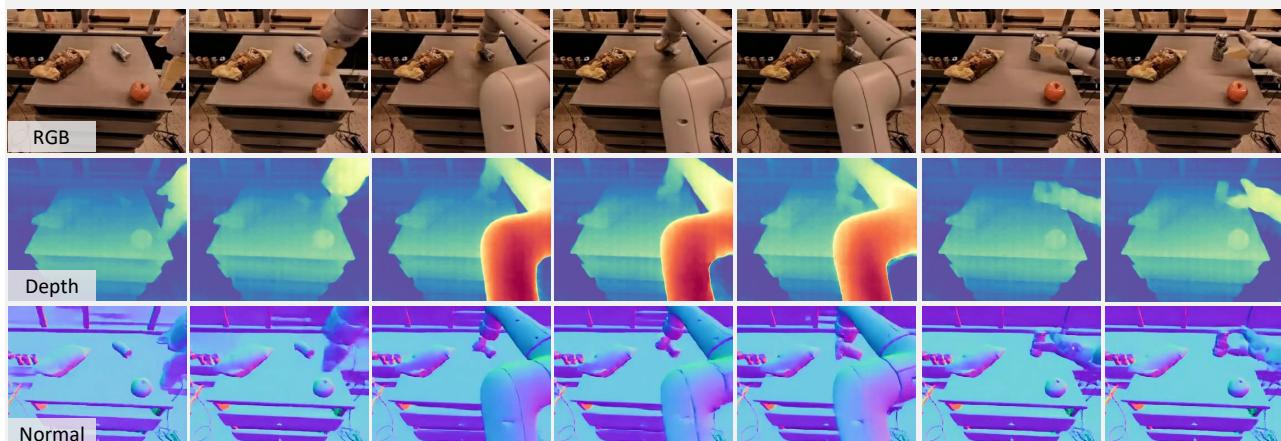Figure 8. Tracking and visualization of robotic arm action trajectories on the Bridge dataset

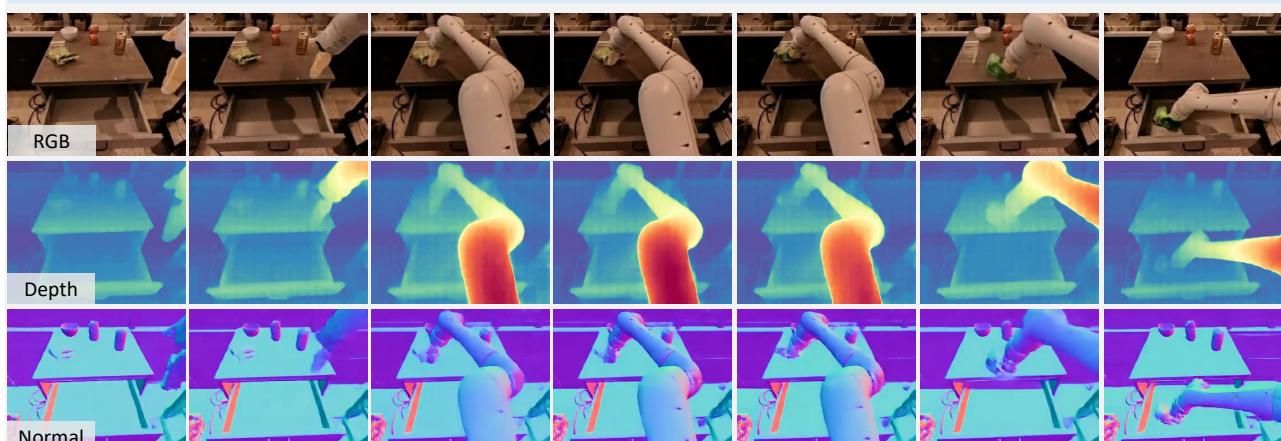Figure 9. In-domain RGB-DN Video generation results on Bridge dataset

Figure 10. In-domain RGB-DN Video generation results on RT1 dataset

Figure 11. In-domain RGB-DN Video generation results on RLBench dataset

Sample Frames from the Datasets.

Bridge: Put stuffed duck in pan.

Bridge: Put pan on stove and put stuffed pig in pan.

RT-1: Place redbull can into middle drawer.
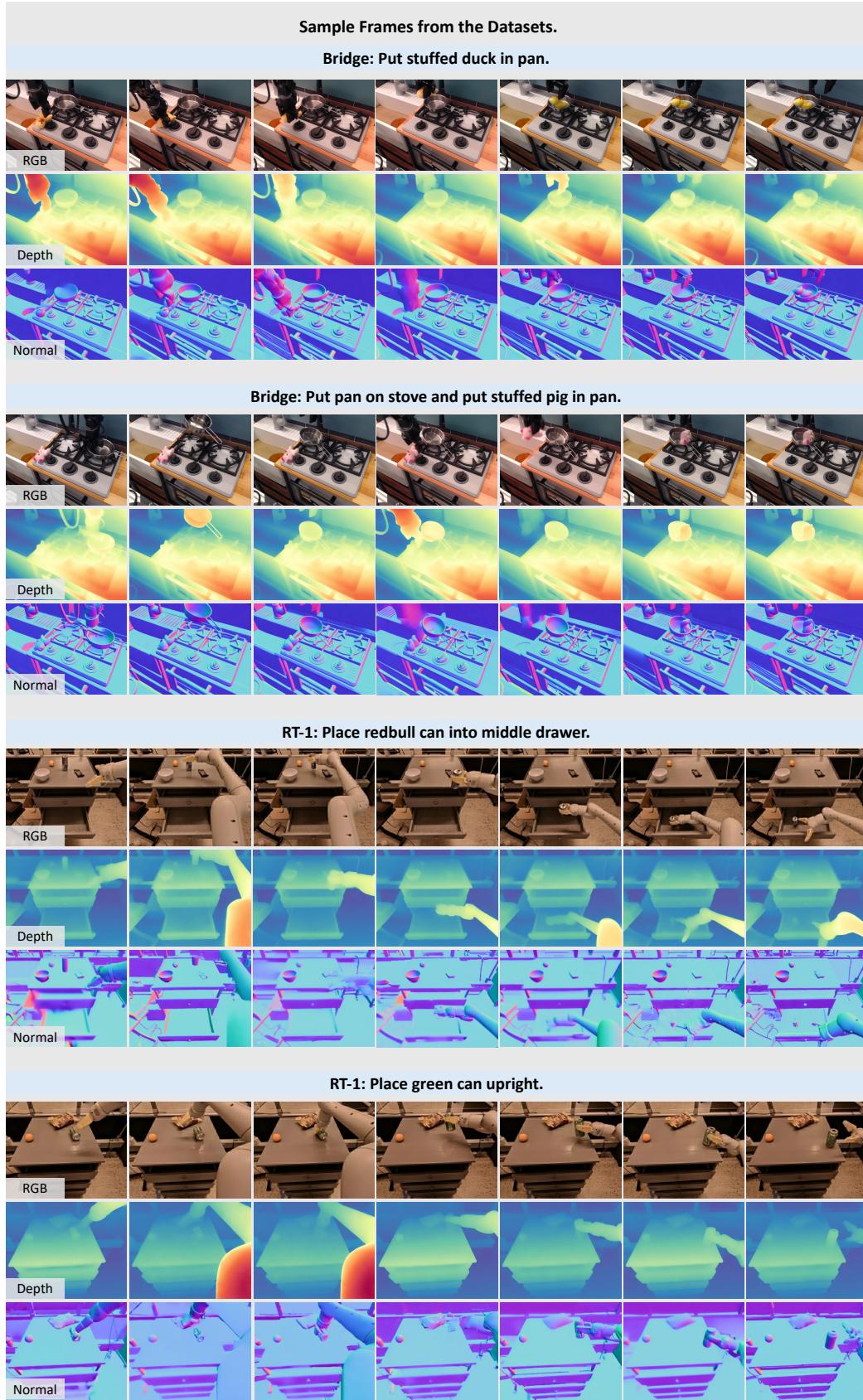
RT-1: Place green can upright.

Figure 12. Some sample frames extracted from the datasets Bridge [61] and RT-1 [6].