Masked Visual Actions for Unified World Modeling

Hadi Alzayer[1,2] **Wenlong Huang**[1] **Haonan Chen**[1,3] **Christopher Luey**[1]

Lvmin Zhang[1] **Maneesh Agrawala**[1] **Gordon Wetzstein**[1] **Li Fei-Fei**[1]

Yilun Du[3] **Jiajun Wu**[1] **Jia-Bin Huang**[2]

[1]Stanford University [2]University of Maryland, College Park [3]Harvard University

https://masked-visual-actions.github.io

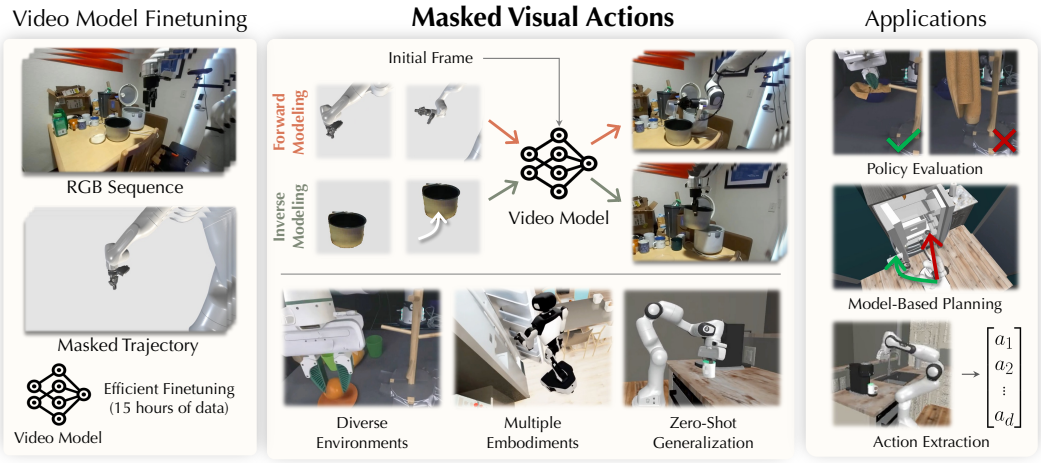

Figure 1: **Masked Visual Actions.** We finetune a video model to condition on masked trajectories of robots, representing robot actions as pixel-space masked motions. Efficiently finetuned on only 15 hours of data, a single checkpoint of the model can act as an action-conditioned forward model to simulate robotic interactions with diverse and unseen embodiments. By conditioning it on object motion, it can also act as an inverse model that synthesizes the robot motion needed to achieve the desired outcome. We showcase the efficacy of our model for policy evaluation, model-based-planning, and action extraction where the video model acts as a policy.

Abstract

Video models absorb rich priors over how the visual world moves, interacts, and responds to contact, making them promising substrates for robotic world modeling. The central challenge is how to communicate action to such models in a form aligned with the visual space in which they learned these interaction priors, yet still grounded in physical manipulation. We introduce **Masked Visual Actions**, a pixel-space control interface that expresses action as a partially revealed trajectory of an arbitrary entity in a video. Revealing robot motion makes the model act as a forward dynamics model that predicts the scene's response to low-level robot actions, while revealing desired object motion makes the same model recover robot behavior consistent with that outcome. Finetuned with only 15 hours of masked examples from real videos and simulation, a single checkpoint achieves strong visual fidelity and controllability across diverse scenes and multiple embodiments. In downstream manipulation settings, the model produces imagined rollouts whose

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

outcomes correlate with real-world execution for policy evaluation, improves decision making by ranking candidate futures in model-based planning, and supports inverse modeling by synthesizing robot motion from desired object motion.

1 Introduction

Purposeful interaction requires connecting an agent's actions to their effects in the world, and vice versa. In sensorimotor control, skilled behavior is often described as coupling a forward model that anticipates the sensory consequences of movement with an inverse model that recovers the movement needed to realize a desired state [68, 69, 70]. Robotic world models should similarly support both directions of reasoning in a single predictive framework.

Recent advances in video models offer a promising route to this ambition. Trained on large-scale observation, they accumulate remarkably broad priors over motion, contact, persistence, deformation, and change, far beyond what can usually be distilled from robot data alone. However, most still remain passive observers rather than tools for intervention. Existing models condition generation on text [53, 64], tracks [13, 20, 57, 59, 83], forces [21, 22, 44], keypoints [67, 71], or motor commands [19, 25], signals that are often sparse, embodiment-specific, or misaligned with the model's pre-trained visual experience. What remains missing is an action representation expressed directly in the visual space where pretrained video models learned their interaction priors. Once action is expressed *visually*, the same model can complete different parts of an interaction depending on the trajectory revealed: revealing robot motion prompts a scene response, while revealing object motion prompts robot behavior.

To realize this vision, we introduce **Masked Visual Actions**, a method that recasts action as a visual primitive directly within the pre-trained model's native representation. We finetune a pre-trained video model [64] to ingest actions as partially revealed spatiotemporal patterns in pixel space—a masked trajectory of an entity in the scene. When the revealed entity is the robot, the model predicts the scene's response and acts as a forward dynamics model; when the revealed entity is instead an object or desired object motion, the same model acts as an inverse model to recover robot behavior consistent with that outcome. In this view, active and passive roles are not properties of separate architectures, but different queries to the same interaction prior. While conceptually simple, this interface is pixel-aligned, embodiment-agnostic, native to video, and efficient to inject into a pretrained model through lightweight adaptation.

In addition to superior visual fidelity compared to prior work, we validate our framework across three applications in robot manipulation, both in simulation and in the real world: policy evaluation, model-based planning, and inverse modeling, where the video model is used as part of a robot policy. All experiments use a *single* checkpoint of the model finetuned on as few as 15 hours of robot interaction data. In policy evaluation, the model's imagined rollouts exhibit consistent correlation with real-world outcomes, so that simulated performance serves as a useful proxy for actual execution. In model-based planning, these same predictive capabilities are used to simulate the effects of different action trajectories and select the best one for execution, leading to consistent gains across diverse tasks and policy architectures. The same checkpoint can also be used in reverse: given desired object motion, it synthesizes robot motion that achieves the goal, and a learned inverse dynamics model extracts the resulting actions.

We summarize our contributions as follows: **(1)** we introduce Masked Visual Actions, a pixel-space control interface for pretrained video models, together with an efficient adaptation recipe based on masked examples from real and simulated data; **(2)** we show that forward and inverse robot world-modeling problems can be cast as complementary conditional prediction problems of the same video model, obtained by revealing different entities in the scene; and **(3)** we validate this framework in both simulation and the real world across three applications in robot manipulation: policy evaluation, model-based planning, and inverse modeling.

2 Related Work

Controllable video generation as a robotic interface. Video models become simulators once a control signal is added: physical forces [21, 22], warped flow [5], hand poses [23, 32, 71], point

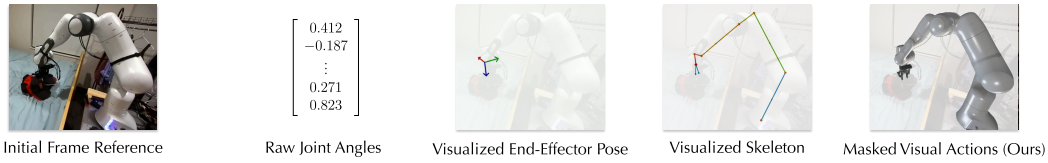

Figure 2: **Comparing action representations for learning.** Low-dimensional robot actions are compact, but embodiment-specific and not aligned with the image observations used by video models. End-effector poses or robot skeletons are more visual, but remain sparse and require the model to infer geometry, contact, and interaction effects. Our masked visual actions provide dense, image-aligned conditioning, making robot motion and action directly visible, yielding a more learnable representation across embodiments and object interaction.

or trajectory tracks [13, 20, 57, 59], and goal images [27]. None of these signals is dense, pixel-aligned, or shareable across embodiments. Closer to us, visual prompting via inpainting [1] and world modeling as conditional inference [35] show that varying which region of an image is provided enables a generic task parametrization in visual domains; we extend this idea to robotic world modeling by using masked frames as the control interface.

Pixel-grounded action conditioning for robot world models. Most robotic video world models communicate actions through embodiment-specific channels: end-effector poses [25, 55, 74, 83], joint vectors [19, 25], or skeletons [61, 67]. A growing line replaces them with pixel-grounded signals. BridgeV2W [10] and Kinema4D [72] render the robot through its URDF and inject the resulting masks or pointmaps via ControlNet; Action Images [80] encodes 7-DoF actions as multi-view Gaussian heatmaps; ORV [75] conditions on 4D occupancy; Mask2IV [40] conditions on predicted mask trajectories; Mask World Model [47] predicts semantic masks as the output. A complementary line treats masks as a data-editing tool: Shadow [9, 36], Phantom [38], Masquerade [37], and EmbodiSwap [14] composite or render the robot onto human videos for cross-embodiment policy transfer. All of these works treat the robot as the active entity during training and run forward only. We expose the same masking interface to *any* subset of entities, so one model serves as a forward, inverse, or unconditional generator without retraining.

Unified video-action models and downstream uses. UVA [42], UWM [82], AIM [18], X-WAM [24], and MotuBrain [50] unify forward dynamics, inverse dynamics, policy, and video generation in one model by masking modality channels (action vector vs. video) or by manipulating diffusion timesteps; large platforms such as Cosmos [53], Genie Envisioner [43], and DreamGen [29] package similar capabilities at foundation-model scale. Because the masking is over modality channels, actions remain low-dimensional vectors and the unification does not transfer across embodiments. Our masking is *spatial*: active and passive entities live on the same pixel canvas, so the same forward/inverse switch also bridges the embodiment gap. The forward direction is exercised for policy evaluation [61, 66, 76], policy improvement [25], planning [7, 16, 17, 28, 56, 62, 79], and direct video-as-policy [26, 33]; the inverse direction extracts robot motion through point tracking [3, 30], object flow [34, 41, 81], predicted object pose [60], or learned IDMs [8, 16, 17, 54, 65, 77]. Critically, our inverse pipeline reuses the same backbone as the forward simulator, while prior work trains a separate IDM head or flow predictor.

3 Masked Visual Actions

Video generation models can be used to show how an initial scene evolves over time [4, 64], as they can model rich scene dynamics and object interactions. The video model captures the distribution $p(V)$ over videos $V \in \mathbb{R}^{T\times H\times W\times 3}$ depicting a scene. We view a scene $\mathcal{S}$ as a set of entities $e_1, e_2, ..., e_n$, and the video model generates a sequence of frames depicting how the entities interact over time. In the output video, each e_i has a spatiotemporal trajectory, and we abuse notation slightly and write e_i for both the entity and the spatiotemporal region of pixels it occupies. Aligned with recent works built on structured masking and condition inference [2, 35, 45, 63], the video model implicitly captures the joint distribution over all entity trajectories

$$p(V) = p(e_1, e_2, \ldots, e_n), \tag{1}$$

including the interactions among them. Conditioning on a subset $\mathcal{S} \subseteq \{1, \ldots, n\}$ of entities yields the conditional distribution

$$p\big(\{e_i\}_{i \notin \mathcal{S}} \,\big|\, \{e_j\}_{j \in \mathcal{S}},\, I_0\big), \tag{2}$$

where I_0 is a reference image of the initial scene. By varying $\mathcal{S}$, the same model answers different questions about the same scene.

We realize this conditioning by masking. Let $M \in \{0,1\}^{T \times H \times W}$ be a binary mask indicating which spatiotemporal pixels are revealed to the model ($M_{t,h,w} = 1$) versus predicted ($M_{t,h,w} = 0$). For a chosen conditioning set $\mathcal{S}$, the mask is the union of the pixel regions occupied by the conditioned entities,

$$M(\mathcal{S}) = \bigcup_{i \in \mathcal{S}} e_i, \tag{3}$$

and the model receives as input the masked video $M \odot V$ together with the reference image I_0. Training proceeds by sampling M from a distribution over masks and learning the conditional $p_\theta(V \mid M \odot V, I_0)$. We draw inspiration from masked modeling in language [15] and masked-image prompting [1], where varying the masked input enables diverse applications with the same model.

In robotics, it is convenient to partition entities into two roles, as illustrated in Figure 3. We call entity e_i *active* if it acts on the scene through its own agency, such as a robot arm or a human, and *passive* if its motion arises from interaction with an active entity, such as a manipulated object. Let $\mathcal{A} \subseteq \{1, \ldots, n\}$ index the active entities and $\mathcal{P} = \{1, \ldots, n\} \setminus \mathcal{A}$ the passive ones. This partition surfaces two natural ways to use the model.

Forward model. Setting $\mathcal{S} = \mathcal{A}$, we condition on the active entities and predict the passive ones,

$$p\big(\{e_i\}_{i \in \mathcal{P}} \,\big|\, \{e_j\}_{j \in \mathcal{A}},\, I_0\big). \tag{4}$$

This corresponds to the standard action-conditioned dynamics modeling, in which a robot's motion is provided, and the model simulates its effect on the scene. Unlike prior work that conditions on low-dimensional action commands [19, 25], the active conditioning here is supplied as masked videos that are agnostic to embodiments.

Inverse model. Setting $\mathcal{S} = \mathcal{P}$, we condition on the passive entities and predict the active ones,

$$p\big(\{e_i\}_{i \in \mathcal{A}} \,\big|\, \{e_j\}_{j \in \mathcal{P}},\, I_0\big). \tag{5}$$

This direction has no analog in conventional action-conditioned world models: the user specifies a desired outcome in the world, and the video model recovers the agent behavior consistent with it.

The active/passive distinction is a convenient way to describe how we use the model rather than a property of the model itself. The model is trained on masked video completion without any explicit notion of agency, and at inference, any subset $\mathcal{S}$ can be chosen. In fact, we trained our model only on masks depicting active robotic entities, yet it generalizes to queries conditioned on passive entities in a *zero-shot* manner. As observed in our empirical evaluations, this behavior is unique to our masked visual action conditioning, as conditioning the video model on sparser signals, such as low-level action commands or visualized skeletons, cannot achieve this level of generalization.

4 Method

4.1 Dataset construction

We construct the masked modeling dataset by combining real-world videos from DROID [31] and simulation data from Robocasa [52]. We use both success and failure trajectories from both datasets. We follow two approaches to construct masked conditioning for each video, based on video segmentation and rendering the robot state, as outlined below.

Segmentation based dataset Given any video, we can segment any entity in the scene using SegmentAnything [6], without the need for camera calibration or even explicitly knowing which robot is shown in the video. We use videos from DROID, and use the prompt "A robotic arm" for segmentation to isolate the robot. Using segmentation data enables the model to effectively learn to *inpaint* missing regions and to model the joint distribution over all entities in the scene. While

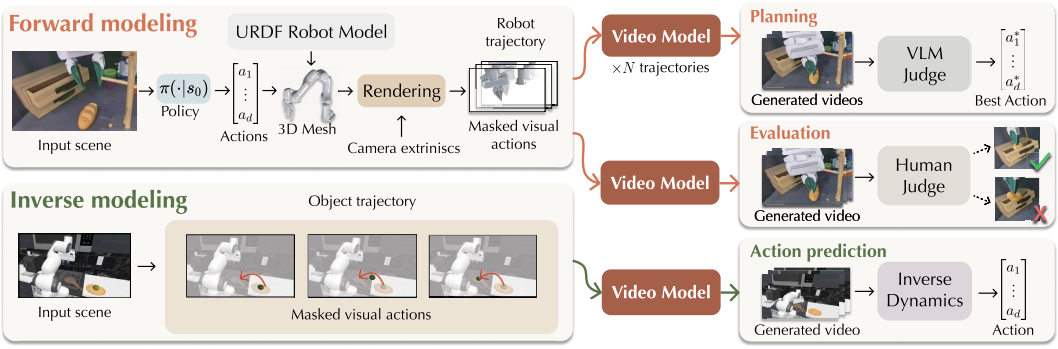

Figure 3: **Applications.** The masked visual actions allow using the video model as a forward model, conditioned on robot actions, or as an inverse model that predicts the robot motion that satisfies the object trajectory. The forward model can be used for planning and choosing the best trajectory sampled from a policy, or policy evaluation. On the other hand, the inverse modeling can be combined with an inverse dynamics model to estimate robot actions from the generated video.

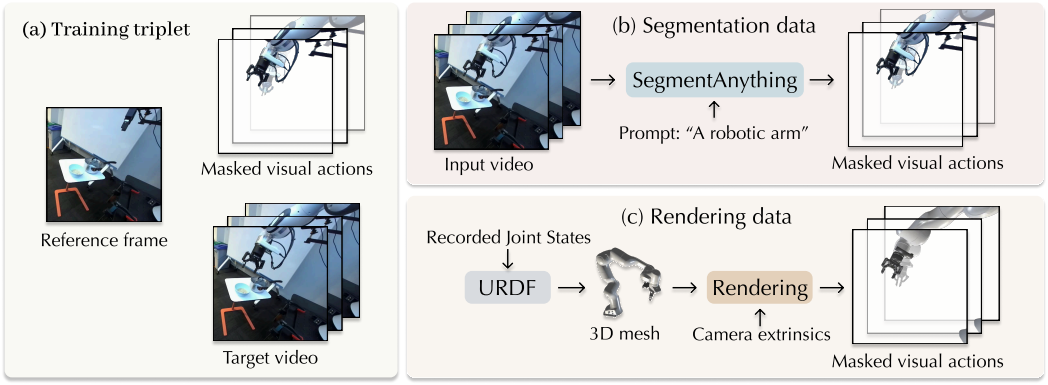

Figure 4: **Dataset construction.** (a) to train our model, we need a reference frame of the initial scene, and the masked visual actions, and train it to reproduce a realistic video of the robot executing the input actions. (b) We use segmentation-based approach by segmenting the robot arm from robotics datasets as the masked visual actions. (c) However, to allow the user to provide arbitrary action trajectory at inference, the model needs to also accept simulated mesh visualization of those actions. As a result, we also include rendering-based dataset. Given that DROID also contains the robot state at each timestep, we render the robot URDF that matches the original video to construct masked visual actions.

a segmentation-based approach is highly general, it suffers from two major limitations: First, it is challenging for the user to provide an exact segmentation mask of entities at test time. Second, any occluded regions in the robot would implicitly leak information about the scene dynamics from the original video. To mitigate those limitations, we also explore the approach that explicitly renders robots from their recorded state.

Rendering based dataset Instead of relying on segmentation, we can also align a robot mesh with the input video and use that as the masked conditioning. By rendering the robot mesh, we can visualize arbitrary action trajectories during inference and then use them as masked conditioning for the video model. To construct a rendering-based dataset, we require the robot state corresponding to the input video and the camera calibration. We use the DROID dataset and follow the protocols from PointWorld [28] to refine the camera calibration to accurately align the robot URDF with the input trajectories. In the Robocasa simulation, we render only the robot, excluding the rest of the scene, to generate the masked conditioning. To allow the model to see the full robot without self-occlusion, we render the robot only with translucent rendering and set the gripper fingers to bright red so the video model can easily observe the actions. Note that the rendering approach requires known camera calibration, and is limited to rendering the robot only as opposed to arbitrarily enabling masking any entity in the scene. As a result, we believe that the segmentation and rendering-based approaches are complementary.

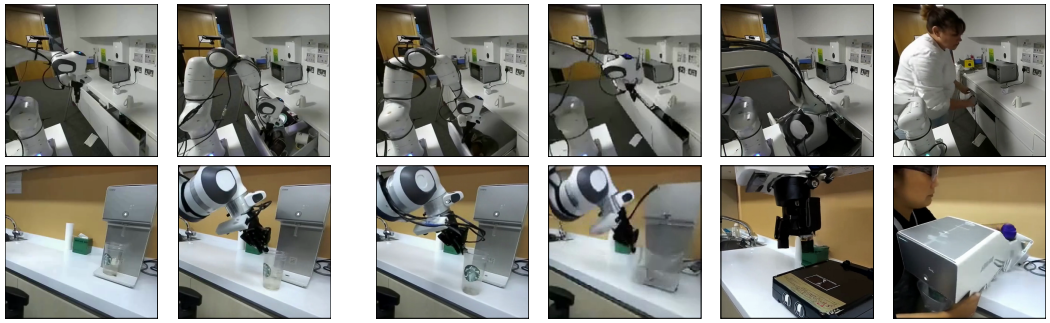

Figure 6: **Comparing baselines on DROID.** Using image-to-video [64] or even trajectory conditioned video generation [13] with GT tracks fails to execute the robot motion or preserve the input scene. On the other hand, our model can competitively match and outperform models that take the raw robot actions [25] while maintaining generalization.

Table 1: **Baseline comparison on diverse embodiments.** We evaluate our method against Ctrl-World [25] on DROID as a seen robotic embodiment, as well as BEHAVIOR, which uses a bimanual robotic embodiment that's unseen for all methods. Our model outperform the baseline on both datasets, and we include image-to-video and trajectory conditioned video models as a reference.

	DROID			BEHAVIOR		
Method	LPIPS ↓	SSIM ↑	PSNR ↑	LPIPS ↓	SSIM ↑	PSNR ↑
Image-to-video [64]	0.521	0.548	12.42	0.602	0.457	10.22
Wan-move [13]	0.534	0.562	12.99	0.312	0.756	13.17
Ctrl-World [25]	0.362	0.708	18.15	0.196	0.837	18.39
Masked Visual Actions (Ours)	**0.0945**	**0.887**	**23.74**	**0.123**	**0.843**	**22.90**

4.2 Model implementation and training

We use Wan-Fun-Control 2.2 14B [64] as the base model. We encode the masked conditioning video using the same autoencoder as the video model and use concatenation as the conditioning mechanism. Concatenation is appropriate as the conditioning signal is spatially aligned with the desired output video. For the missing region from the masked conditioning, we set it to a uniform gray background. Instead of finetuning the entire model, we use LoRA finetuning with rank 256, and a batch size of 4 using 8 NVIDIA H200 GPUs. We train the model for approximately 10,000 steps over 4 days. *For reproducibility, we will release our code, data, and model weights.*

5 Experiments

We start by evaluating Masked Visual Actions as a control signal for world modeling. We evaluate visual fidelity and controllability against prior work and highlight generalization to embodiments unseen during training. Afterward, we evaluate diverse robotic applications of our video model. In particular, we show how it can be used for planning by evaluating sampled trajectories, for policy evaluation, and for using the video model as the policy itself through inverse modeling. *Please refer to the project webpage for video results.*

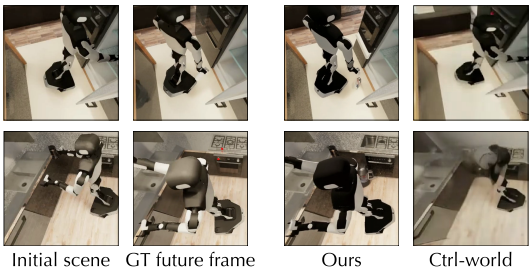

Figure 5: **Generalization to unseen embodiment.** While using the raw action state such as in Ctrl-world [25] can work well within the training domain, it collapses on unseen embodiments.

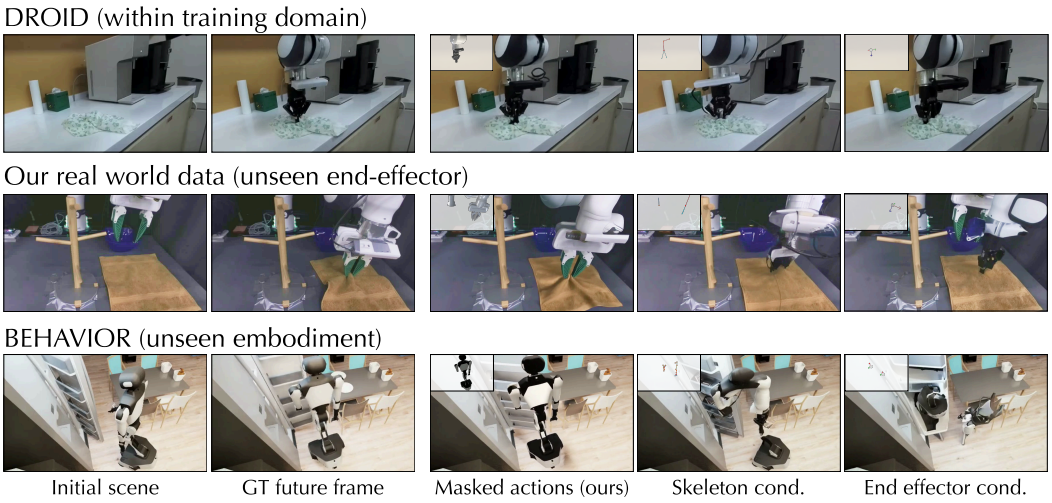

Figure 7: **Comparing action conditioning.** Training a video model on different conditioning signals on DROID such as masked visual actions, end effector visualization, or skeleton all work well within the training domain. However, when going beyond the training distribution, such as using a custom end-effector, the models trained on skeleton and end effector position would hallucinate the robot seen in training or transform the robot to match the training. Furthermore, on unseen embodiments such as bimanual robots in BEHAVIOR, using masked actions generalizes gracefully, while other conditioning signals transform and disfigure the robot.

Table 2: **Ablation on visual conditioning signal.** When using sparse conditioning signal, the performance on held out data from the training distribution on DROID is similar to using masked visual actions. However, when using the same robot with unseen gripper (such as on our real world data), or on a robot from unseen embodiment in BEHAVIOR, the gap increases significantly between our masked actions and the other conditioning methods.

	DROID			Real world			BEHAVIOR		
Method	LPIPS ↓	SSIM ↑	PSNR ↑	LPIPS ↓	SSIM ↑	PSNR ↑	LPIPS ↓	SSIM ↑	PSNR ↑
End-effector vis.	0.107	0.878	22.64	0.183	0.858	20.32	0.171	0.815	19.23
Skeleton vis.	0.106	0.878	22.74	0.169	**0.866**	21.02	0.162	0.824	19.58
Masked Visual Actions	**0.0945**	**0.887**	**23.74**	**0.148**	0.864	**22.79**	**0.123**	**0.843**	**22.90**

5.1 Controllable video generation

Video generation traditionally conditioned on text is expressive, but underspecified. Conditioning on motion tracks preserves the generality while allowing us to condition on motion. On the other hand, conditioning on the action space for a specific robotic embodiment provides additional precision, but at the cost of generality. Through Masked Visual Actions, we aim to preserve the generality of the video model by visually conditioning the video model on the robotic actions, and setting the role of the video model to answer: *given this visual masked action, how would the rest of the scene look like?* As a baseline for using the robot's raw end-effector state as input, we use Ctrl-world [25], a recent SoTA method. For track conditioning, we use Wan-move [13], conditioning it on ground-truth tracks computed from the robot mesh. Additionally, we include Wan2.2 14B image-to-video as a reference.

In Fig. 6, we highlight that both Masked Visual Actions and Ctrl-world accurately follow the robot actions on our held out scenes from DROID[1] On the other hand, both Wan-Move and Wan I2V completely collapse and transform the input scene. However, unlike our method, conditioning on the raw robot state cannot generalize to unseen embodiments [39, 73]. We use data from BEHAVIOR [39], which uses a bimanual robot, R1-Pro, to evaluate generalization on unseen embodiments. In Fig. 5, we demonstrate that Ctrl-world simply outputs static or corrupted videos for unseen embodiments, while our model can gracefully handle the unseen embodiment. We quantitatively evaluate performance on generated videos across both DROID and BEHAVIOR in Table 1 and show that our method outperforms the baselines.

[1] Ctrl-World was trained on the entirety of DROID, so it has seen our held out scenes during training.

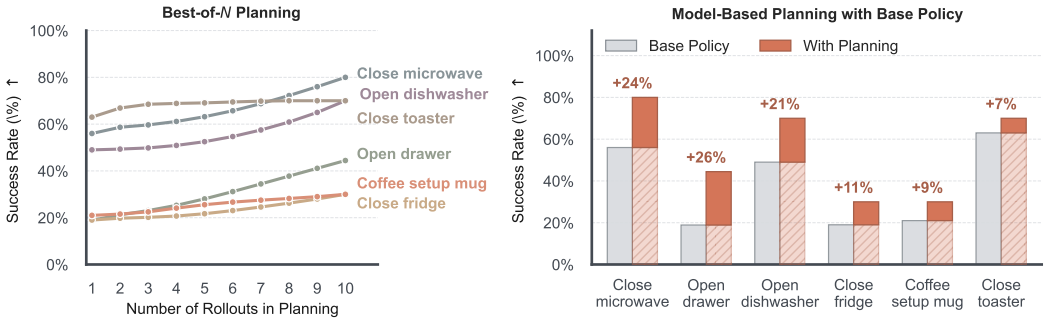

Figure 8: **Application on Planning.** By rolling out multiple trajectories from a pretrained diffusion policy, we can evaluate each trajectory by simulating the actions with the video model, and then using a VLM judge to pick the best action trajectory. We observe consistent improvement in task success when using the video model to roll out and choose best action sequences, as well as the positive correlation with the number of samples evaluated at test time. This demonstrates the ability of the model to simulate counterfactuals given the same initial condition.

Comparing the choice of visual actions Conditioning on visual actions allows diverse ways to represent the action. We compare against visualizing the end-effector pose, inspired by IRASim [83], and the robot skeleton, adopted in VAP [67]. We train the same base model used for our method, and use the same training dataset from DROID to train the baselines. While we expect the varying action conditioning to perform similarly on the same domain as the training set, sparse conditioning signals require the model to explicitly learn the correspondence between the sparse action and the target video. However, by conditioning on masked visual actions, the model simply needs to model the interaction between the masked input and the rest of the scene. In Fig. 7, we show that on DROID, all the variants of our model perform similarly. However, on real-world data we captured using a similar robot to the one used in DROID, the Franka Emika Panda, but with a custom 3D-printed end-effector, we find that using a sparse conditioning signal suffers significantly. In particular, when conditioning on the robot skeleton, the video model would transform the robot to match the embodiment seen during training. When using the end-effector visualization as input, the model would simply introduce another robot into the scene that matches the training data. To further deviate from the training setting, we test the models on the R1 Pro in BEHAVIOR [39]. Given that R1 Pro has two end effectors, we adapt the baseline visualizations to show two end effectors and the skeleton poses of each. We find that conditioning on the end-effector or skeleton visualization completely collapses and distorts the robot. However, when using masked visual actions, the model can gracefully generalize and simulate the physical interaction of the robot opening the fridge. In Table 2, we quantitatively evaluate the video generation performance for masked visual actions, and the sparser conditioning mechanisms of end effector pose visualization and robot skeleton.

5.2 Robotics applications

We highlight multiple applications of our unified world model in robotics. We use our model as a forward model to simulate robot actions and demonstrate its use for planning and policy evaluation. We also use our model as an inverse model: given the desired object motion as a masked visual action, we generate a video of the robot performing the desired object manipulation and extract the actions using a learned inverse dynamics model. Across these applications, we use Robocasa [52] as the simulation environment.

Planning Given the same environment observation, multiple rollouts sampled from a stochastic policy may achieve varying levels of task progress. By rolling them out in an action-conditioned video model, one may evaluate the trajectories purely in imagination before executing them in the actual environment. In our experiments, we use Diffusion Policy [11, 12] as the stochastic policy and Best-of-N as the simplest model-based planning algorithm. After simulating the action candidates using the video model, we evaluate each rollout with Gemini 3.1 Pro to assess their relative task success, interaction fidelity, and physical realism. We evaluate on 10 scenes per task, with $N = 10$. We include the detailed criteria in the appendix. After evaluating all the rollouts, we pick the best action sequence to execute. In Fig. 8, we highlight the improvement in performance on diverse tasks and show how success rate increases with the number of action samples. This approach can be viewed as a form of test-time scaling [48, 51], leveraging additional compute to achieve higher performance. In our case, the policy and video model act as the generator, and the VLM critic acts as the verifier.

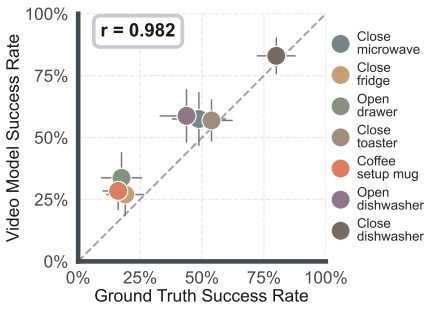

Figure 9: **Robocasa policy evaluation.** Video-model rollouts consistently track ground-truth success rates across RoboCasa tasks.

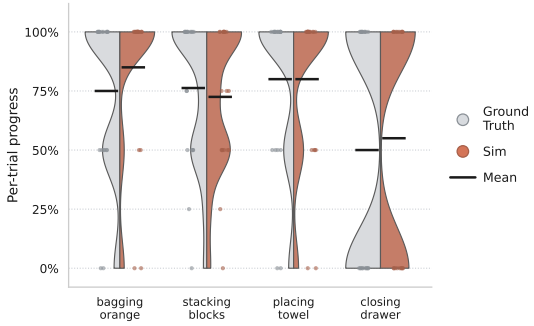

Figure 10: **Real world policy evaluation.** Rolling out real-world demonstrations with our video model produces videos with success progress closely aligned with what is observed in the real world execution.

Policy evaluation We can also use our model to evaluate policy performance by comparing a policy's success rate within the video model to that within the ground-truth environment. Similarly to the section above, we use an open-loop diffusion policy across tasks and rollout 10 trajectories per scene. We simulate action trajectories using our model and manually evaluate each rollout as a success or failure based on predefined task rubrics. The simulated rollouts are additionally evaluated by physical interaction realism (e.g., hallucinated task progress without contact is considered failure). In Fig. 9, we plot the success rate of each policy in GT environment against that evaluated within the video model, which exhibits a strong correlation with $r = 0.982$. However, we observe that the video model shows a positive bias towards task progress, as evidenced by consistently higher task success rates in its imagination. Beyond simulation, we evaluate our model in a real-world setup. For each of four tasks we collect 20 demonstrations, roll out each demonstration with the video model, and score both the real and simulated executions with a per-task rubric measuring partial task progress. For each task, we collect 20 demonstrations, and set a rubric for evaluating the success progress for each demonstration. Because each simulated rollout is paired with the demonstration it was generated from, we compare progress both in distribution and per trial. In Fig. 10 we plot the per-trial progress distribution in the video model against the ground-truth distribution for each task. The two distributions closely match, but similar to simulation, it shows a positive bias towards task progress. We include each task's rubric in the appendix and all generated videos and GT demonstrations in the project webpage.

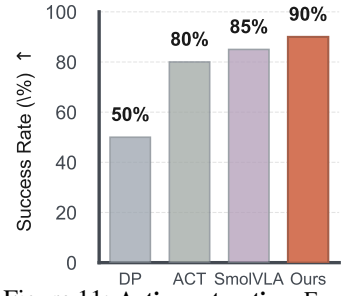

Figure 11: **Action extraction.** Even without task-specific video-model training, inverse modeling recovers competitive robot behavior.

Action extraction Instead of providing the model with robot actions to execute, we can alternatively leverage Masked Visual Actions to formulate an inverse modeling problem: given desired object motion, prompting the video model to synthesize a video of a robot achieving that outcome. Action extraction can then be cast as an inverse-dynamics problem: given the synthesized robot video, recover an executable low-level action sequence with a learned inverse-dynamics model. We initially expected this setting to require explicit inverse-modeling finetuning. Instead, the video model trained only on forward examples already generalizes zero-shot to the inverse setting, likely because the conditioning signal is well-aligned with the model's learned representation. We evaluate this pipeline on CoffeeServeMug in RoboCasa, where the robot must reach, grasp, transport, and place a mug from the coffee machine onto the table in a tightly constrained workspace. We compare to standard imitation learning baselines, including Diffusion Policy [11, 12], ACT [78], and SmolVLA [58]. The inverse-dynamics model and all baselines are trained on 100 demonstrations, whereas the video model itself has not seen examples from this task. Each method is evaluated with 20 trials, with success rates reported in Figure 11. Our method achieves the highest success rate at 90%. This

indicates that, although the video model has not been trained on an inverse modeling problem, it can be effectively prompted to extract robot behaviors from its rich interaction priors, while maintaining the competitiveness of modern imitation learning methods.

6 Discussion and Conclusions

By finetuning a pretrained video model on a small amount of Masked Visual Actions data, we efficiently leverage the prior of the video model to synthesize counterfactuals by conditioning on a subset of scene entities. Our model can simulate robot actions when conditioned on robotic embodiment visualization as an action-conditioned forward model, and when acting as an inverse model, where it synthesizes suitable robot motion to realistically manipulate the object.

Limitations It is worth noting that our model, similarly to existing generative models, learns the *correlation* between object interaction rather than *causal relationships*, which remains an open research question. Furthermore, our method is naturally limited by the base video model's capabilities, in terms of both inference speed and what it can express, as it re-purposes the model's prior rather than modifying its capabilities.

Societal and broader impact By enabling video-based policy evaluation, planning, and inverse modeling, our work could lower the cost of developing robotic systems and make robot learning more accessible. However, the same capabilities could also be used for unsafe or unauthorized robotic behaviors, highlighting the need for responsible use and deployment.

Acknowledgments and Disclosure of Funding

We would like to thank Stephen Tian, Zeyu Chen, and Wensi Ai for the fruitful discussions. This paper is in part funded by ONR MURI N00014-22-1-2740 and ONR MURI N00014-24-1-2748, with computing equipment in part supported by the NVIDIA Academic Grant Program.

References

[1] Amir Bar, Yossi Gandelsman, Trevor Darrell, Amir Globerson, and Alexei A. Efros. Visual prompting via image inpainting. In *NeurIPS*, 2022.

[2] Daniel M Bear, Kevin Feigelis, Honglin Chen, Wanhee Lee, Rahul Venkatesh, Klemen Kotar, Alex Durango, and Daniel LK Yamins. Unifying (machine) vision via counterfactual world modeling. *arXiv preprint arXiv:2306.01828*, 2023.

[3] Homanga Bharadhwaj, Roozbeh Mottaghi, Abhinav Gupta, and Shubham Tulsiani. Track2act: Predicting point tracks from internet videos enables generalizable robot manipulation, 2024.

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

[5] Ryan Burgert, Yuancheng Xu, Wenqi Xian, Oliver Pilarski, Pascal Clausen, Mingming He, Li Ma, Yitong Deng, Lingxiao Li, Mohsen Mousavi, Michael Ryoo, Paul Debevec, and Ning Yu. Go-with-the-flow: Motion-controllable video diffusion models using real-time warped noise. In *CVPR*, 2025.

[6] Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris Coll-Vinent, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, Jie Lei, Tengyu Ma, Baishan Guo, Arpit Kalla, Markus Marks, Joseph Greer, Meng Wang, Peize Sun, Roman Rädle, Triantafyllos Afouras, Effrosyni Mavroudi, Katherine Xu, Tsung-Han Wu, Yu Zhou, Liliane Momeni, RISHI HAZRA, Shuangrui Ding, Sagar Vaze, Francois Porcher, Feng Li, Siyuan Li, Aishwarya Kamath, Ho Kei Cheng, Piotr Dollar, Nikhila Ravi, Kate Saenko, Pengchuan Zhang, and Christoph Feichtenhofer. SAM 3: Segment anything with concepts. In *ICLR*, 2026.

[7] Boyuan Chen, Tianyuan Zhang, Haoran Geng, Kiwhan Song, William T. Freeman, Jitendra Malik, Russ Tedrake, Vincent Sitzmann, and Yilun Du. Large video planner, 2025.

[8] Haonan Chen, Jiaming Xu, Lily Sheng, Tianchen Ji, Shuijing Liu, Yunzhu Li, and Katherine Driggs-Campbell. Learning coordinated bimanual manipulation policies using state diffusion and inverse dynamics models. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025.

[9] Haonan Chen, Cheng Zhu, Shuijing Liu, Yunzhu Li, and Katherine Rose Driggs-Campbell. Tool-as-interface: Learning robot policies from observing human tool use. In *Proceedings of Robotics: Conference on Robot Learning (CoRL)*, 2025.

[10] Yixiang Chen, Peiyan Li, Jiabing Yang, Keji He, Xiangnan Wu, Yuan Xu, Kai Wang, Jing Liu, Nianfeng Liu, Yan Huang, et al. Bridgev2w: Bridging video generation models to embodied world models via embodiment masks. *arXiv preprint arXiv:2602.03793*, 2026.

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

[12] 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*, 2024.

[13] Ruihang Chu, Yefei He, Zhekai Chen, Shiwei Zhang, Xiaogang Xu, Bin Xia, Dingdong Wang, Hongwei Yi, Xihui Liu, Hengshuang Zhao, et al. Wan-move: Motion-controllable video generation via latent trajectory guidance. *arXiv preprint arXiv:2512.08765*, 2025.

[14] Eadom Dessalene, Pavan Mantripragada, Michael Maynord, and Yiannis Aloimonos. Embodiswap for zero-shot robot imitation learning, 2025.

[15] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423.

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

[17] Yilun Du, Sherry Yang, Pete Florence, Fei Xia, Ayzaan Wahid, Pierre Sermanet, Tianhe Yu, Pieter Abbeel, Joshua B Tenenbaum, Leslie Pack Kaelbling, et al. Video language planning. In *The Twelfth International Conference on Learning Representations*, 2023.

[18] Liaoyuan Fan, Zetian Xu, Chen Cao, Wenyao Zhang, Mingqi Yuan, and Jiayu Chen. Aim: Intent-aware unified world action modeling with spatial value maps, 2026.

[19] Shenyuan Gao, William Liang, Kaiyuan Zheng, Ayaan Malik, Seonghyeon Ye, Sihyun Yu, Wei-Cheng Tseng, Yuzhu Dong, Kaichun Mo, Chen-Hsuan Lin, Qianli Ma, Seungjun Nah, Loic Magne, Jiannan Xiang, Yuqi Xie, Ruijie Zheng, Dantong Niu, You Liang Tan, K.R. Zentner, George Kurian, Suneel Indupuru, Pooya Jannaty, Jinwei Gu, Jun Zhang, Jitendra Malik, Pieter Abbeel, Ming-Yu Liu, Yuke Zhu, Joel Jang, and Linxi "Jim" Fan. Dreamdojo: A generalist robot world model from large-scale human videos. *arXiv preprint arXiv:2602.06949*, 2026.

[20] Daniel Geng, Charles Herrmann, Junhwa Hur, Forrester Cole, Serena Zhang, Tobias Pfaff, Tatiana Lopez-Guevara, Carl Doersch, Yusuf Aytar, Michael Rubinstein, Chen Sun, Oliver Wang, Andrew Owens, and Deqing Sun. Motion prompting: Controlling video generation with motion trajectories. In *CVPR*, 2025.

[21] Nate Gillman, Charles Herrmann, Michael Freeman, Daksh Aggarwal, Evan Luo, Deqing Sun, and Chen Sun. Force prompting: Video generation models can learn and generalize physics-based control signals. In *NeurIPS*, 2025.

[22] Nate Gillman, Yinghua Zhou, Zitian Tang, Evan Luo, Arjan Chakravarthy, Daksh Aggarwal, Michael Freeman, Charles Herrmann, and Chen Sun. Goal force: Teaching video models to accomplish physics-conditioned goals. In *CVPR*, 2026.

[23] Raktim Gautam Goswami, Amir Bar, David Fan, Tsung-Yen Yang, Gaoyue Zhou, Prashanth Krishnamurthy, Michael Rabbat, Farshad Khorrami, and Yann LeCun. World models for learning dexterous hand-object interactions from human videos. *arXiv preprint arXiv:2512.13644*, 2026.

[24] Jun Guo, Qiwei Li, Peiyan Li, Zilong Chen, Nan Sun, Yifei Su, Heyun Wang, Yuan Zhang, Xinghang Li, and Huaping Liu. Unified 4d world action modeling from video priors with asynchronous denoising, 2026.

[25] Yanjiang Guo, Lucy Xiaoyang Shi, Jianyu Chen, and Chelsea Finn. Ctrl-world: A controllable generative world model for robot manipulation. In *ICLR*, 2026.

[26] 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, 2024.

[27] Siqiao Huang, Jialong Wu, Qixing Zhou, Shangchen Miao, and Mingsheng Long. Vid2world: Crafting video diffusion models to interactive world models, 2025.

[28] Wenlong Huang, Yu-Wei Chao, Arsalan Mousavian, Ming-Yu Liu, Dieter Fox, Kaichun Mo, and Fei-Fei Li. Pointworld: Scaling 3d world models for in-the-wild robotic manipulation. In *CVPR*, 2026.

[29] Joel Jang, Seonghyeon Ye, Zongyu Lin, Jiannan Xiang, Johan Bjorck, Yu Fang, Fengyuan Hu, Spencer Huang, Kaushil Kundalia, Yen-Chen Lin, Loic Magne, Ajay Mandlekar, Avnish Narayan, You Liang Tan, Guanzhi Wang, Jing Wang, Qi Wang, Yinzhen Xu, Xiaohui Zeng, Kaiyuan Zheng, Ruijie Zheng, Ming-Yu Liu, Luke Zettlemoyer, Dieter Fox, Jan Kautz, Scott Reed, Yuke Zhu, and Linxi Fan. Dreamgen: Unlocking generalization in robot learning through video world models, 2025.

[30] Nikita Karaev, Iurii Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker3: Simpler and better point tracking by pseudo-labelling real videos, 2024.

[31] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, Peter David Fagan, Joey Hejna, Masha Itkina, Marion Lepert, Yecheng Jason Ma, Patrick Tree Miller, Jimmy Wu, Suneel Belkhale, Shivin Dass, Huy Ha, Arhan Jain, Abraham Lee, Youngwoon Lee, Marius Memmel, Sungjae Park, Ilija Radosavovic, Kaiyuan Wang, Albert Zhan, Kevin Black, Cheng Chi, Kyle Beltran Hatch, Shan Lin, Jingpei Lu, Jean Mercat, Abdul Rehman, Pannag R Sanketi, Archit Sharma, Cody Simpson, Quan Vuong, Homer Rich Walke, Blake Wulfe, Ted Xiao, Jonathan Heewon Yang, Arefeh Yavary, Tony Z. Zhao, Christopher Agia, Rohan Baijal, Mateo Guaman Castro, Daphne Chen, Qiuyu Chen, Trinity Chung, Jaimyn Drake, Ethan Paul Foster, Jensen Gao, Vitor Guizilini, David Antonio Herrera, Minho Heo, Kyle Hsu, Jiaheng Hu, Muhammad Zubair Irshad, Donovon Jackson, Charlotte Le, Yunshuang Li, Kevin Lin, Roy Lin, Zehan Ma, Abhiram Maddukuri, Suvir Mirchandani, Daniel Morton, Tony Nguyen, Abigail O'Neill, Rosario Scalise, Derick Seale, Victor Son, Stephen Tian, Emi Tran, Andrew E. Wang, Yilin Wu, Annie Xie, Jingyun Yang, Patrick Yin, Yunchu Zhang, Osbert Bastani, Glen Berseth, Jeannette Bohg, Ken Goldberg, Abhinav Gupta, Abhishek Gupta, Dinesh Jayaraman, Joseph J Lim, Jitendra Malik, Roberto Martín-Martín, Subramanian Ramamoorthy, Dorsa Sadigh, Shuran Song, Jiajun Wu, Michael C. Yip, Yuke Zhu, Thomas Kollar, Sergey Levine, and Chelsea Finn. Droid: A large-scale in-the-wild robot manipulation dataset. 2024.

[32] Byungjun Kim, Taeksoo Kim, Junyoung Lee, and Hanbyul Joo. Dexterous world models. In *CVPR*, 2026.

[33] Moo Jin Kim, Yihuai Gao, Tsung-Yi Lin, Yen-Chen Lin, Yunhao Ge, Grace Lam, Percy Liang, Shuran Song, Ming-Yu Liu, Chelsea Finn, and Jinwei Gu. Cosmos policy: Fine-tuning video models for visuomotor control and planning, 2026.

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

[35] Klemen Kotar, Wanhee Lee, Rahul Venkatesh, Honglin Chen, Daniel Bear, Jared Watrous, Simon Kim, Khai Loong Aw, Lilian Naing Chen, Stefan Stojanov, et al. World modeling with probabilistic structure integration. *arXiv preprint arXiv:2509.09737*, 2025.

[36] Marion Lepert, Ria Doshi, and Jeannette Bohg. Shadow: Leveraging segmentation masks for cross-embodiment policy transfer, 2025.

[37] Marion Lepert, Jiaying Fang, and Jeannette Bohg. Masquerade: Learning from in-the-wild human videos using data-editing, 2025.

[38] Marion Lepert, Jiaying Fang, and Jeannette Bohg. Phantom: Training robots without robots using only human videos, 2025.

[39] Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Wensi Ai, Benjamin Martinez, Hang Yin, Michael Lingelbach, Minjune Hwang, Ayano Hiranaka, Sujay Garlanka, Arman Aydin, Sharon Lee, Jiankai Sun, Mona Anvari, Manasi Sharma, Dhruva Bansal, Samuel Hunter, Kyu-Young Kim, Alan Lou, Caleb R Matthews, Ivan Villa-Renteria, Jerry Huayang Tang, Claire Tang, Fei Xia, Yunzhu Li, Silvio Savarese, Hyowon Gweon, C. Karen Liu, Jiajun Wu, and Li Fei-Fei. Behavior-1k: A human-centered, embodied ai benchmark with 1,000 everyday activities and realistic simulation. *arXiv preprint arXiv:2403.09227*, 2024.

[40] Gen Li, Bo Zhao, Jianfei Yang, and Laura Sevilla-Lara. Mask2iv: Interaction-centric video generation via mask trajectories, 2025.

[41] Hongyu Li, Lingfeng Sun, Yafei Hu, Duy Ta, Jennifer Barry, George Konidaris, and Jiahui Fu. Novaflow: Zero-shot manipulation via actionable flow from generated videos, 2025.

[42] Shuang Li, Yihuai Gao, Dorsa Sadigh, and Shuran Song. Unified video action model, 2025.

[43] Yue Liao, Pengfei Zhou, Siyuan Huang, Donglin Yang, Shengcong Chen, Yuxin Jiang, Yue Hu, Jingbin Cai, Si Liu, Jianlan Luo, Liliang Chen, Shuicheng Yan, Maoqing Yao, and Guanghui Ren. Genie envisioner: A unified world foundation platform for robotic manipulation, 2025.

[44] Wei Liu, Ziyu Chen, Zizhang Li, Yue Wang, Hong-Xing Yu, and Jiajun Wu. Realwonder: Real-time physical action-conditioned video generation, 2026.

[45] Khai Loong Aw, Klemen Kotar, Wanhee Lee, Seungwoo Kim, Khaled Jedoui, Rahul Venkatesh, Lilian Naing Chen, Michael C Frank, and Daniel LK Yamins. Zero-shot world models are developmentally efficient learners. *arXiv e-prints*, pages arXiv–2604, 2026.

[46] Ilya Loshchilov, Frank Hutter, et al. Fixing weight decay regularization in adam. *arXiv preprint arXiv:1711.05101*, 5(5):5, 2017.

[47] Yunfan Lou, Xiaowei Chi, Xiaojie Zhang, Zezhong Qian, Chengxuan Li, Rongyu Zhang, Yaoxu Lyu, Guoyu Song, Chuyao Fu, Haoxuan Xu, Pengwei Wang, and Shanghang Zhang. Mask world model: Predicting what matters for robust robot policy learning, 2026.

[48] Nanye Ma, Shangyuan Tong, Haolin Jia, Hexiang Hu, Yu-Chuan Su, Mingda Zhang, Xuan Yang, Yandong Li, Tommi Jaakkola, Xuhui Jia, and Saining Xie. Inference-time scaling for diffusion models beyond scaling denoising steps. 2025.

[49] Ajay Mandlekar, Soroush Nasiriany, Bowen Wen, Iretiayo Akinola, Yashraj Narang, Linxi Fan, Yuke Zhu, and Dieter Fox. Mimicgen: A data generation system for scalable robot learning using human demonstrations. *arXiv preprint arXiv:2310.17596*, 2023.

[50] MotuBrain Team, Chendong Xiang, Fan Bao, Haitian Liu, Hengkai Tan, Hongzhe Bi, James Li, Jiabao Liu, Jingrui Pang, Kiro Jing, Louis Liu, Mengchen Cai, Rongxu Cui, Ruowen Zhao, Runqing Wang, Shuhe Huang, Yao Feng, Yinze Rong, Zeyuan Wang, and Jun Zhu. Motubrain: An advanced world action model for robot control, 2026.

[51] Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng, editors, *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 20275–20321, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1025.

[52] Soroush Nasiriany, Sepehr Nasiriany, Abhiram Maddukuri, and Yuke Zhu. Robocasa365: A large-scale simulation framework for training and benchmarking generalist robots. In *ICLR*, 2026.

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

[54] Jonas Pai, Liam Achenbach, Victoriano Montesinos, Benedek Forrai, Oier Mees, and Elvis Nava. mimic-video: Video-action models for generalizable robot control beyond vlas, 2025.

[55] Han Qi, Haocheng Yin, Aris Zhu, Yilun Du, and Heng Yang. Inference-time enhancement of generative robot policies via predictive world modeling. *IEEE Robotics and Automation Letters*, 2026.

[56] Yichao Shen, Fangyun Wei, Zhiying Du, Yaobo Liang, Yan Lu, Jiaolong Yang, Nanning Zheng, and Baining Guo. Videovla: Video generators can be generalizable robot manipulators. In *NeurIPS*, 2025.

[57] Joonghyuk Shin, Zhengqi Li, Richard Zhang, Jun-Yan Zhu, Jaesik Park, Eli Shechtman, and Xun Huang. MotionStream: Real-Time Video Generation with Interactive Motion Controls. In *ICLR*, 2026.

[58] Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, et al. Smolvla: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025.

[59] Assaf Singer, Noam Rotstein, Amir Mann, Ron Kimmel, and Or Litany. Time-to-move: Training-free motion-controlled video generation via dual-clock denoising. In *ICLR*, 2026.

[60] Yue Su, Xinyu Zhan, Hongjie Fang, Yong-Lu Li, Cewu Lu, and Lixin Yang. Motion before action: Diffusing object motion as manipulation condition, 2024.

[61] Gemini Robotics Team, Krzysztof Choromanski, Coline Devin, Yilun Du, Debidatta Dwibedi, Ruiqi Gao, Abhishek Jindal, Thomas Kipf, Sean Kirmani, Isabel Leal, Fangchen Liu, Anirudha Majumdar, Andrew Marmon, Carolina Parada, Yulia Rubanova, Dhruv Shah, Vikas Sindhwani, Jie Tan, Fei Xia, Ted Xiao, Sherry Yang, Wenhao Yu, and Allan Zhou. Evaluating gemini robotics policies in a veo world simulator, 2025.

[62] Rhoda AI Team. Causal video models are data-efficient robot policy learners. *Rhoda AI Blog*, 2026.

[63] Rahul Venkatesh, Honglin Chen, Kevin Feigelis, Daniel M Bear, Khaled Jedoui, Klemen Kotar, Felix Binder, Wanhee Lee, Sherry Liu, Kevin A Smith, et al. Understanding physical dynamics with counterfactual world modeling. In *European Conference on Computer Vision*, pages 368–387. Springer, 2024.

[64] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

[65] Ruixiang Wang, Qingming Liu, Yueci Deng, Guiliang Liu, Zhen Liu, and Kui Jia. Eva: Aligning video world models with executable robot actions via inverse dynamics rewards, 2026.

[66] Yixuan Wang, Rhythm Syed, Fangyu Wu, Mengchao Zhang, Aykut Onol, Jose Barreiros, Hooshang Nayyeri, Tony Dear, Huan Zhang, and Yunzhu Li. Interactive world simulator for robot policy training and evaluation. *arXiv preprint arXiv:2603.08546*, 2026.

[67] Yuang Wang, Chao Wen, Haoyu Guo, Sida Peng, Minghan Qin, Hujun Bao, Xiaowei Zhou, and Ruizhen Hu. Precise action-to-video generation through visual action prompts. In *ICCV*, 10 2025.

[68] Daniel M. Wolpert and J. Randall Flanagan. Motor prediction. *Current Biology*, 11(18): R729–R732, 2001. doi: 10.1016/S0960-9822(01)00432-8.

[69] Daniel M. Wolpert, Zoubin Ghahramani, and Michael I. Jordan. An internal model for sensorimotor integration. *Science*, 269(5232):1880–1882, 1995. doi: 10.1126/science.7569931.

[70] Daniel M. Wolpert, R. Chris Miall, and Mitsuo Kawato. Internal models in the cerebellum. *Trends in Cognitive Sciences*, 2(9):338–347, 1998. doi: 10.1016/S1364-6613(98)01221-2.

[71] Linxi Xie, Lisong C. Sun, Ashley Neall, Tong Wu, Shengqu Cai, and Gordon Wetzstein. Generated reality: Human-centric world simulation using interactive video generation with hand and camera control. *arXiv preprint arXiv:2602.18422*, 2026.

[72] Mutian Xu, Tianbao Zhang, Tianqi Liu, Zhaoxi Chen, Xiaoguang Han, and Ziwei Liu. Kinema4d: Kinematic4d world modeling for spatiotemporal embodied simulation. *arXiv preprint arXiv:2603.16669*, 2026.

[73] Xiaomeng Xu, Dominik Bauer, and Shuran Song. RoboPanoptes: The All-Seeing Robot with Whole-body Dexterity. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.042.

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

[75] Xiuyu Yang, Bohan Li, Shaocong Xu, Nan Wang, Chongjie Ye, Zhaoxi Chen, Minghan Qin, Yikang Ding, Zheng Zhu, Xin Jin, Hang Zhao, and Hao Zhao. Orv: 4d occupancy-centric robot video generation, 2025.

[76] Kaifeng Zhang, Shuo Sha, Hanxiao Jiang, Matthew Loper, Hyunjong Song, Guangyan Cai, Zhuo Xu, Xiaochen Hu, Changxi Zheng, and Yunzhu Li. Real-to-sim robot policy evaluation with gaussian splatting simulation of soft-body interactions. In *ICRA*, 2026.

[77] Zhongru Zhang, Chenghan Yang, Qingzhou Lu, Yanjiang Guo, Jianke Zhang, Yucheng Hu, and Jianyu Chen. Veo-act: How far can frontier video models advance generalizable robot manipulation?, 2026.

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

[79] 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.

[80] Haoyu Zhen, Zixian Gao, Qiao Sun, Yilin Zhao, Yuncong Yang, Yilun Du, Pengsheng Guo, Tsun-Hsuan Wang, Yi-Ling Qiao, and Chuang Gan. Action images: End-to-end policy learning via multiview video generation, 2026.

[81] Hongyan Zhi, Peihao Chen, Siyuan Zhou, Yubo Dong, Quanxi Wu, Lei Han, and Mingkui Tan. 3dflowaction: Learning cross-embodiment manipulation from 3d flow world model, 2025.

[82] Chuning Zhu, Raymond Yu, Siyuan Feng, Benjamin Burchfiel, Paarth Shah, and Abhishek Gupta. Unified world models: Coupling video and action diffusion for pretraining on large robotic datasets, 2025.

[83] Fangqi Zhu, Hongtao Wu, Song Guo, Yuxiao Liu, Chilam Cheang, and Tao Kong. Irasim: Learning interactive real-robot action simulators. In *ICCV*, 10 2025.

Table B1: Reconstruction metrics across datasets (mean ± SEM; best per column in bold).

	BEHAVIOR-1K (n=50)			DROID (n=50)			Real-World (n=13)		
Method	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Ctrl-World	18.39±0.54	0.836±0.008	0.196±0.011	18.15±0.16	0.708±0.008	0.362±0.006	—	—	—
Wan-Move	13.17±0.30	0.528±0.015	0.454±0.016	12.96±0.37	0.558±0.017	0.534±0.023	—	—	—
Wan2.2-I2V	10.22±0.38	0.457±0.023	0.602±0.024	12.42±0.34	0.548±0.017	0.521±0.023	—	—	—
EEF	19.23±0.34	0.815±0.006	0.171±0.007	22.64±0.26	0.878±0.005	0.107±0.005	20.32±0.25	0.858±0.005	0.183±0.008
Skeleton	19.58±0.34	0.824±0.006	0.162±0.007	22.74±0.28	0.878±0.005	0.106±0.004	21.02±0.38	**0.866±0.005**	0.169±0.010
Ours	**22.90±0.52**	**0.842±0.007**	**0.123±0.007**	**23.74±0.24**	**0.887±0.004**	**0.095±0.004**	**22.79±0.35**	0.864±0.008	**0.148±0.010**

Appendix Overview

This appendix provides detailed supplementary material supporting the main paper on video-based world models for robotic manipulation. It is organized into eight sections covering implementation details, evaluation protocols, and data collection procedures:

1. **Reproducibility**: Commitment to release complete code, model weights, and training data to enable reproduction of results.
2. **Full quantitative results**: Comprehensive tables presenting quantitative metrics across all reconstruction experiments with standard error bars, demonstrating consistent improvements over baselines.
3. **Additional training details**: Dataset composition and construction, including use of DROID (1,000 demonstrations) and Robocasa (4,000 examples), handling of failure cases, and conditioning strategy for the video model.
4. **VLM Evaluation Protocol**: Automated evaluation methodology using Gemini 3.1 Pro Preview to assess generated robot videos. Details the Gemini system prompt used, task-agnostic failure detection (ghost contact, post-disengagement coasting, frame-jump glitches), structured JSON output format, rollout ranking via lexicographic ordering, and subset enumeration for success-rate curves.
5. **Policy and Inverse-Dynamics Training Details**: Training procedures for three policy settings—simulation policies for planning/evaluation in RoboCasa, inverse-dynamics model for action extraction from generated videos, and real-world policies for physical robot studies. Covers Diffusion Policy, ACT, and SmolVLA baselines.
6. **Robot Data Collection**: Hardware setup, calibration procedures, and data representation. Describes Franka Panda manipulator with custom end-effector, dual ZED Mini 2 cameras (wrist-mounted and external), AprilTag-based hand-eye calibration, and HDF5 trajectory storage with camera intrinsics and extrinsics.
7. **Adapting Baselines to Unseen Embodiments**: Methodology for evaluating generalization to unseen robot morphologies using BEHAVIOR-1K benchmark. Describes adaptation strategies for the video model (direct URDF conditioning), Ctrl-World baseline (heuristic-based active arm identification), and skeleton/pose visualizations for bimanual robots.
8. **Robustness to Conditioning Quality**: Robustness experiments studying the effect of frame dropping and pixel/patch occlusion in the masked visual action conditioning on reconstruction quality.

A Reproducibility

To ensure full reproducibility, we will release our code, model weights, and training data.

B Full quantitative results

For completeness, in Table B1 we present the quantitative results across all reconstruction experiments, along with the standard error for each metric. Our method consistently outperforms all the baselines, and within the standard error of conditioning on the skeleton for the real-world data we captured using an unseen custom gripper on the robot.

C Additional training details

In DROID, to ensure that our validation and testing do not include any scenes from the training data, we exclude all scenes from the labs CLVR and RAD from the training set and reserve them for our test set. We use approximately 1,000 DROID demonstrations with the two external cameras and process each demonstration with both the segmentation-based and rendering-based pipelines. We ensure that we keep the failure cases from DROID to ensure that the model can generate counterfactuals accurately. We also include 4,000 examples from Robocasa across all tasks that do not involve robot navigation, and generate failure cases to incorporate in the training data. Since the base video model requires a prompt as a condition, we include only a high-level prompt to ensure the model relies solely on the visual conditioning signal for video generation.

D VLM Evaluation Protocol

In the policy evaluation, we used an off-the-shelf VLM to assess the generated videos, fully automating the process. Our evaluation protocol is as follows: For each generated rollout video, we query Gemini 3.1 Pro Preview (`gemini-3.1-pro-preview`, temperature 0, sampled at 15 fps so all 81 frames of the clip are seen) with a task-agnostic system instruction that calls out three world-model failure modes, the evaluator must penalize: ghost/weak contact (end-effector near but not touching), post-disengagement coasting (object continues toward the goal after the end-effector retracts), and frame-jump glitches (goal state appears after a teleport). The model returns a structured JSON object with four scored fields: `robot_caused_outcome` (boolean `caused`, categorical `mechanism` $\in$ { `robot_pushed`, `robot_grasped`, `teleported`, `vanished`, `autonomous`, `passthrough`, `no_attempt`, `other`}), `task_success` (boolean plus confidence), `end-effector_contact` (boolean plus target object), and `physics_realism` (1–5 score with an `issues` list). Field order is enforced via the response schema so that the causal mechanism is committed *before* the binary success flag, anchoring the verdict on process rather than the final frame. Task-specific success criteria (e.g. "the toaster door must be flush, latched, and closed by a push from below") are passed in the per-rollout user prompt rather than encoded in the system instruction.

To select one rollout per scene from N_s candidates, we sort by the lexicographic key $\kappa = (\mathbf{1}[\text{success}], \text{realism}, \text{confidence}, \mathbf{1}[\text{contact}], -r)$ in descending order: prefer success-flagged rollouts; among them, prefer higher physics realism; then higher confidence; then visible contact; tiebreak deterministically by lower rollout index r. The pick is $\arg\max_r \kappa(v_{s,r})$, and the planner's per-task success rate is the fraction of scenes for which the picked rollout is GT-correct (`max_reward` $= 1.0$). To plot success as a function of N, we exhaustively enumerate every size-N subset of the scene's rollouts, take Gemini's top-ranked rollout within each subset, and average the resulting hit rate across the $\binom{N_s}{N}$ subsets and across scenes. The random baseline reduces to this expression at $N{=}1$, where the planner has no choice and merely returns the per-scene mean GT rate.

D.1 Gemini system prompt

The system instruction below is sent verbatim with every video query (model: `gemini-3.1-pro-preview`, temperature 0, sampled at 15 fps). Per-rollout user prompts add only the task description and the task-specific success criterion; all evaluation policy lives in this system instruction.

```
You are a strict evaluator of robot-manipulation videos.
The video was produced by a generative video model that is meant to
*simulate* a robot policy executing a task. The video is often
visually flawed and may HALLUCINATE the desired outcome -- e.g. the
target object teleports into its goal pose, vanishes and reappears in
place, or moves on its own without the robot causing it. Your job is
to judge whether the *robot* actually caused the outcome via
plausible physical contact, not whether the final frame happens to
look correct.

CRITICAL ANTI-PATTERNS -- these are common world-model artifacts that
look superficially like success but must NOT be credited as
robot_pushed / robot_grasped:
```

```
  (a) Ghost / weak contact. The gripper must be in visible mechanical
      coupling with the object on the frames the object moves: the
      gripper geometry should be touching/compressing against the
      object surface, and the object's motion should be co-located in
      time with that contact. If the gripper is merely *near* the
      object, in front of it, or has only momentary glancing contact
      and the object still moves, treat the motion as autonomous or
      passthrough -- NOT as robot_pushed. "Implied" or "magic" contact
      does not count.

  (b) Post-disengagement coasting / "momentum". The simulated
      environment has NO momentum or follow-through: an object does
      not continue moving toward its goal after the gripper retracts
      or stops. If the gripper disengages (retracts, stops, or moves
      away) and the object continues to move closed/into-goal on its
      own afterwards, the post-disengagement frames are autonomous,
      regardless of what came before. Only credit the extent of
      progress that occurred WHILE the gripper was in active contact
      AND moving the object. If the goal state is only reached after
      the gripper has disengaged, robot_caused_outcome.caused is
      FALSE and task_success.success is FALSE.

  (c) Glitchy reach-around or frame jumps. If the goal state appears
      in a frame following a visual discontinuity, scene reset, or
      sudden teleport -- even if the robot's pose looks plausible
      around it -- treat as teleported.

Score these fields:

  1. robot_caused_outcome
       caused      True only if the robot, via its gripper, is the
                   visible physical cause of the goal-relevant motion,
                   AND the goal state is reached DURING the contact
                   window (not after it).
       mechanism   one of:
                     "robot_pushed"      gripper pushed/pulled the object
                                         in sustained contact
                     "robot_grasped"     gripper grasped and moved it
                     "teleported"        object snapped to goal state
                     "vanished"          object disappeared from scene
                     "autonomous"        object moved on its own
                                         (incl. post-disengagement
                                         coasting, ghost contact)
                     "passthrough"       gripper went through the object
                     "no_attempt"        robot never engaged the object
                     "other"             describe in reasoning
  2. task_success      success=TRUE requires ALL THREE of: (a) the
                       end-state matches the success criterion, (b)
                       robot_caused_outcome.caused is true, and (c)
                       the goal state was reached during a frame
                       range where the gripper was actively in
                       mechanical contact with the object. Outcomes
                       reached after disengagement, via ghost contact,
                       teleport, or autonomous motion are FALSE.
  3. gripper_contact   Did the gripper visibly make physical contact
                       with the target object? "Hovering near",
                       "passing through", or single-frame glancing
                       contact does not count.
  4. physics_realism   1-5 plausibility. Penalize teleporting,
                       morphing, gripper passing through solids,
                       vanishing or duplicated objects, frame-to-frame
                       jumps, ghost contact, post-disengagement
                       coasting.

Be terse but specific. Reference what you see in the video (frame
ranges, objects, motions, whether the gripper was in contact when
the goal state was reached). Do not speculate beyond what is
```

```
visible.
```

Per-rollout user message. On top of the system prompt, the per-video user message provides only the natural-language task and the success criterion:

```
Intended task: {task_prompt}
Success criterion: {expected_outcome}
Score physics_realism on a 1-5 integer scale: 1 = nonsensical,
3 = obvious artifacts but recognizable, 5 = indistinguishable from
real video. Return ONLY the JSON object matching the response schema.
```

Structured response schema. The model is constrained to return JSON with five top-level fields, in this order (enforced via `propertyOrdering`): `robot_caused_outcome` (booleans `caused`, categorical `mechanism`, free-form `reasoning`), `task_success` (`success`, `confidence`, `reasoning`), `gripper_contact` (`contact`, `target_object`, `reasoning`), `physics_realism` (integer `score` $\in [1, 5]$, `issues` list, `reasoning`), and a free-form `summary`. Ordering causal mechanism *before* the binary success flag forces the model to commit to a process explanation rather than rationalize from the final frame.

E Policy and Inverse-Dynamics Training Details

Here, we discuss the training details for policy learning and the inverse-dynamics model. We distinguish between three settings: *(i)* simulation policies used for planning and policy evaluation in RoboCasa, *(ii)* the learned inverse-dynamics model used for action extraction, and *(iii)* policies trained for real-world rollout studies.

E.1 Simulation Policy Training for Planning and Policy Evaluation

Unless otherwise stated, the policy used in our planning and policy-evaluation experiments is a Diffusion Policy [11, 12] trained on task-specific mixtures of human demonstrations and MimicGen [49] trajectories provided by the official RoboCasa [52] sources. Each policy conditions on RGB observations from two fixed cameras together with the robot state, and predicts the full 12-D action sequence for the episode horizon. Concretely, these 12 dimensions are 3 end-effector position commands, 3 end-effector rotation commands, 1 end-effector-close command, 3 mobile-base commands, 1 torso command, and 1 control-mode command. Across tasks, the human teleoperation data contain approximately 100 demonstrations, while the MimicGen augmentation expands the training set to roughly thousands of trajectories, depending on the task and horizon. For example, in the MG+Human setting used for our atomic-task studies, the merged training sets are typically on the order of 9×10^3 trajectories.

For the simulation policies, we evaluate them against the success conditions provided by the benchmark. In planning, these policies serve as proposal generators: we sample action candidates from the policy, simulate each candidate with the video model, and then rank the predicted futures with the VLM-based evaluator. In policy evaluation, the same class of policies provides the action trajectories whose success rate is compared between the simulator and the video model.

E.2 Learned Inverse Dynamics for Action Extraction

For action extraction, the video model first synthesizes a robot interaction conditioned on desired object motion, and a learned inverse-dynamics model then converts the generated video into an executable low-level action sequence. This inverse-dynamics model is trained on CoffeeServeMug using the same native 12-D RoboCasa action space. The model predicts the full action sequence with a transformer-based architecture and is optimized with AdamW [46] for 300 epochs.

We also include Diffusion Policy [11, 12], ACT [78], and SmolVLA [58] baselines in this experiment, but only as narrow controls for the action-extraction setting. Those baselines operate on the first

frame of the generated video rather than the full synthesized interaction, and therefore do not require a separate dedicated discussion outside this subsection.

E.3 Real-World Policy Training

For the real-world rollout studies, we additionally train image-conditioned policies directly on two captured manipulation tasks: *mug on rack* and *close cabinet*. These datasets are sampled at 10 Hz and contain 40 trajectories for each task. Each example provides four synchronized RGB views together with an 8-D joint-and-end-effector action sequence. For both tasks, we train Diffusion Policy [11, 12] and ACT [78] policies under the same observation-action contract.

F Robot Data Collection

We collect real-robot data using a Franka Panda manipulator equipped with a custom 3D printed compliant end-effector. Demonstrations are collected through teleoperation with a custom GELLO interface. During each rollout, the operator controls the robot from an initial scene state s_0, while we record synchronized robot states and RGB observations from both wrist-mounted and external cameras.

F.1 Hardware Setup

The sensing setup consists of two ZED Mini 2 cameras. One camera is mounted on the robot's wrist and provides an egocentric view near the end-effector. The second camera is mounted externally and provides a fixed third-person view of the workspace. Both cameras record RGB frames at 720p resolution and 30 Hz.

F.2 Camera and Robot Frame Calibration

We calibrate the camera and robot frames using a table-mounted AprilTag. The AprilTag remains fixed in the workspace throughout calibration. To estimate its pose in the robot base frame, we move the Franka through a set of diverse poses and capture images of the tag from the wrist camera. For each pose, Franka kinematics provide the end-effector frame ($T_E^B(q_t)$), and AprilTag detection provides the tag pose in the wrist-camera frame. These observations are used to estimate the wrist-camera hand-eye transform $T_{C^{\text{wrist}}}^E$ and the AprilTag pose T_A^B in the robot base frame.

The external camera is then calibrated by observing the same table-mounted AprilTag. Since T_A^B is known from the wrist-camera calibration, the external-camera pose relative to the robot base can be computed from the detected tag pose:

$$T_{C^{\text{ext}}}^B = T_A^B \left(T_A^{C^{\text{ext}}}\right)^{-1}.$$

Thus, the table-mounted AprilTag provides a shared reference frame between the robot, wrist camera, and external camera. We verify calibration quality by projecting the robot model into the camera images and checking their alignment with the observed frames.

F.3 Trajectory Representation

Each rollout is stored as a trajectory

$$\tau = \{(I_t^{\text{ext}}, I_t^{\text{wrist}}, q_t)\}_{t=1}^T,$$

where I_t^{ext} and I_t^{wrist} are the external and wrist RGB observations, and q_t is the measured robot joint state.

All data are stored in HDF5 format. Each file corresponds to one rollout and contains RGB frames, joint states, timestamps, and calibration metadata. The calibration metadata includes

$$K_{\text{ext}}, \quad K_{\text{wrist}}, \quad T_A^B, \quad T_{C^{\text{ext}}}^B, \quad T_{C^{\text{wrist}}}^E.$$

For experiments involving visual robot masks, the stored joint states are combined with the Franka kinematic model, custom end-effector geometry, and calibrated camera parameters to render the robot geometry into the image plane.

G Adapting Baselines to Unseen Embodiments

To evaluate generalization to unseen embodiments, we evaluate our model and baseline methods in BEHAVIOR-1K [39], where a bimanual humanoid robot is tasked to complete household mobile manipulation tasks. Since our video model simply takes a masked visual action as input, adapting our model to unseen embodiments that are significantly different from what it has observed is straightforward, as we pass in the rendering of the new robot's URDF. In Ctrl-World, since it is trained to predict multi-view video outputs, we adapt the camera view settings in BEHAVIOR to match DROID's setup. For action conditioning, since Ctrl-World is trained on single-arm robots from DROID, while BEHAVIOR data uses a bimanual robot with a different action space, we first use a heuristic to identify the active arm based on the magnitude of the relative end-effector transformation. Then, we use the identified arm as the action conditioning for Ctrl-World. We find this to be an effective heuristic, as a majority of robot behaviors in the dataset consist of independent arm motions. For the model conditioned on the visualization of the end effector pose, we include both end effectors in the conditioning video. Similarly, for the model conditioned on the robotic arm skeleton, we visualize the skeletons of each robot arm from the bimanual robot.

H Robustness to Conditioning Quality

As we point out earlier, our framework effectively solves the problem of providing visually aligned action conditioning. To further illustrate the resilience of our approach to the quality of the mask conditioning, we conduct robustness experiments where we drop a significant rate of the conditioning frames and introduce varying types of occlusion to the conditioning. Results are summarized in Table H2.

Frame dropping We randomly omit frames of the masked visual actions (Table H2a), and show that the degradation is small and gradual until we reach a drop rate of 80% where PSNR drops $\sim$2dB. In conclusion, this shows that our model is largely robust to missing frames up to 50% dropout.

Robot occlusion: pixel dropout We randomly drop pixels of our masked visual actions (Table H2b), and we show virtually no degradation in performance up to 20% pixel dropout, and minor drop in 30% and 50% of pixel dropout.

Robot occlusion: patch dropout Here, we drop contiguous patches from our masked visual actions (Table H2c), and similarly find that the model is highly robust to occlusion rate up to 20% with no degradation in performance. The model degrades by roughly 1dB when the occlusion rate is 30%.

Table H2: Robustness to conditioning quality (mean ± SEM).

(a) Frame dropping

Drop rate	PSNR↑	SSIM↑	LPIPS↓
0% (clean)	22.26±0.98	0.853±0.013	0.131±0.013
10%	21.80±1.07	0.847±0.014	0.133±0.013
20%	21.81±1.07	0.846±0.014	0.132±0.013
35%	21.59±1.00	0.844±0.013	0.138±0.012
50%	21.36±1.08	0.841±0.014	0.140±0.015
80%	20.24±1.34	0.825±0.015	0.163±0.015

(b) Robot occlusion: pixel dropout

Occlusion rate	PSNR↑	SSIM↑	LPIPS↓
0% (clean)	22.26±0.98	0.853±0.013	0.131±0.013
5%	22.05±1.05	0.848±0.014	0.132±0.014
10%	22.02±0.97	0.848±0.014	0.134±0.012
20%	22.00±1.03	0.847±0.015	0.135±0.013
30%	21.83±1.07	0.846±0.014	0.136±0.012
50%	21.66±1.11	0.843±0.015	0.139±0.012

(c) Robot occlusion: patch dropout

Occlusion rate	PSNR↑	SSIM↑	LPIPS↓
0% (clean)	22.26±0.98	0.853±0.013	0.131±0.013
5%	22.17±1.12	0.852±0.014	0.131±0.013
10%	22.05±1.11	0.850±0.015	0.132±0.017
20%	22.16±1.02	0.850±0.014	0.130±0.013
30%	21.22±0.96	0.838±0.013	0.144±0.011
50%	20.78±0.45	0.835±0.015	0.150±0.014

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: We include experiments highlighting the applications of our model and how it compares to prior work.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

Justification: We discuss the limitations of our method, and include failure cases in supplementary webpage.

Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [N/A]

   Justification:We do not make theoretical claims.

   Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We will be releasing our full code, data, and model weights to ensure reproducibility.

   Guidelines:

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

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

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

   Answer: [Yes]

   Justification: All our code and data will be open sourced.

   Guidelines:

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

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: We explain our training setting and datasets, and highlight the data used for evaluation.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: We provide a table in the supplementary section with the explicit standard error for the tables presented in the paper.

   Guidelines:

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We include the details of the training in the paper.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.

- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We follow the code of conduct.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: We include a societal impact statement.

    Guidelines:

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

11. **Safeguards**

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

    Answer: [Yes]

    Justification: We acknowledge the societal impact of the work, and believe the current model does not include risks that warrant safeguards.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: all content used is open sourced

    Guidelines:

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

13. **New assets**

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

    Answer: [Yes]

    Justification:The primary new asset is the finetuned video model which will be open sourced.

    Guidelines:

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

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

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

Answer: [Yes]

Justification: We manually conduct the experiments requiring human decisions rather than crowdsourcing.

Guidelines:

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

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

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

Answer: [N/A]

Justification: No crowdsourcing experiments were needed

Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [Yes]

Justification: LLMs were used as coding agents, for text editing, and discussing drafting the text.

Guidelines:

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