7. Appendix

7.1. Disclaimer

This is a research work where the primary focus is introducing a new task and a method to learn effective multimodal representation for generative simulation. We devise our multimodal feature extraction as generic to be combined when stronger video generation backbone is invented. High-resolution videos are **not** the main focus of this work. We **provide higher resolution** results of our model in Sec. 7.8.11, and we conduct all experiments shown in the paper using the same video resolution, including our model and all baseline methods trained. We hope our work can inspire future research works and industrial efforts to build foundational digital twin of our world with fine-grained control. We hope that our work can be used to scale with more abundant resources.

7.2. Notation Chart

We summarize the notation used in our paper in Table. 4.

7.3. Related Work

Learning Multi-Modal Representations. Learning shared representations across various modalities has been instrumental in a variety of research areas. Early research by De Sa et al. [12] pioneered the exploration of correlations between vision and audio. Since then, many deep learning techniques have been proposed to learn shared multi-modal representations, including vision-language [14, 29, 41, 53], audio-text [3], vision-audio [4, 27, 45, 46, 49], vision-touch [37, 70], and sound with Inertial Measurement Unit (IMU) [8]. Recently, ImageBind [19] and LanguageBind [76] demonstrate that images and text could successfully bind multiple modalities, including audio, depth, thermal, and IMU, into a shared representation. However, these previous efforts take bind-all fuse-all perspective, which takes away many of the inherent differences brought by various sensory modalities. Our work takes a different perspective. By differentiating between the active and passive senses, we allow a bilateral model to arise and capture the interaction between the two. The prior fuse-all strategy also overshadows an inherent need in multi-modal representation learning, which is interaction. We propose a representation learning scheme to capture the nature of multi-modal interactions.

Learning World Models. Learning accurate dynamics models to predict environmental changes from control inputs has long challenged system identification [39], model-based reinforcement learning [61], and optimal control [5, 77]. Most approaches learn separate lower-dimensional state space models per system instead of directly modeling the high-dimensional pixel space [2, 6, 18, 35]. While simplifying modeling, this limits cross-system knowledge sharing. Recent large transformer architectures enable learning image-based world models, but mostly in visually simplistic, data-abundant simulated games/environments [7, 21, 22, 43, 58, 68]. Prior generative video modeling works leverage text prompts [72, 75], driving motions [60, 66], 3D geometries [67, 69], physical simulations [11], frequency data [38], and user annotations [24] to introduce video movements. Recently, Yang et al. [71] proposes Unisim, which uses text conditioned video diffusion model as an interactive visual world simulator. However, these prior works focus on using text as condition to control video generation, which limits their ability to precisely control the generated video output, as many fine-grained interactions and subtle variations in control are difficult to be accurately described only using text. We propose to use complementary multi-sensory data to achieve more fine-grained temporal control over video generation through multi-sensory action conditioning.

7.4. Implementation Details

Network Architecture Detail We use the open-source I2VGen [74] video diffusion network as our backbone. We

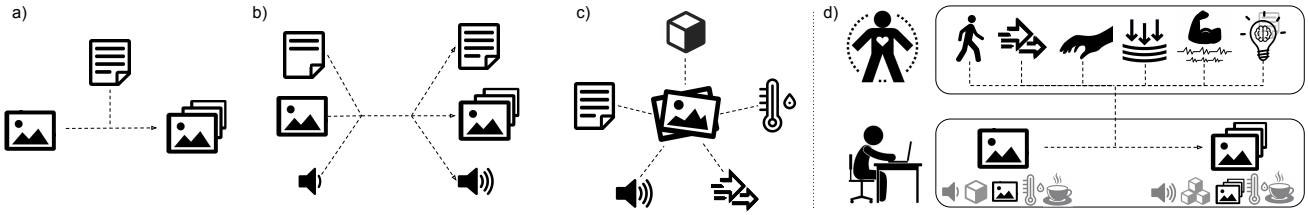

Figure 10. Existing multimodal learning tasks focus on vision-language binding, cross-modal retrieval, and modalitiy anchoring focuses on mining the similarity between different modalities of data (a, b, c) [19, 57, 71]. On the other hand, the task of multisensory action conditioned generative simulation (d) need to understand the unique aspect of each interoceptive action modalities (top) and combine the synchronously to change the exteroception of the external world (bottom).

modify original I2VGen to take pixel space data by changing the input channel to 3 (originally set to 4) and change input image size to 64×64. We keep all other parameters unmodified, and vary the input condition type. We note that single condition models that only use image **or** text such as Stable Diffusion [56] and etc. are not sufficient for our purpose.

All text input are encoded using CLIP text encoder from the open-source OpenClip [1] libary. Images are encoded also using OpenClip Image encoder. Specifically, we use the *ViT-H-14* version with *laion2b_s32b_b79k* weights. Please refer to the original papers [1, 74] their architecture details. We describe the architecture of the remaining modules of our model.

Signal specific encoder heads for hand pose, body pose, emg uses the same MLP architecture with different input dimension. The input dimension for hand pose is $24 \times 3 \times 8$, body pose is $28 \times 3 \times 8$, emg is $8\times$, hand force is $32 \times 32 \times 8$. MLP is composed of four layers, with GeLU activation. We set the hidden and output dimension of 128. We apply a dropout with p=0.1, with batchnorm applied in the first two layers. All encoded signals then goes through a three-layer MLP projection head to project the encoded feature to the same space $\mathbb{R}^{1024}$ as the clip image feature. The projection MLP also uses GeLU activation with dimensions of [input_dim, 512, 768, 1024]. We apply batchnorm after the first layer. The set of features are then aggregated across the sensory modalities and masked by a softmax in the modality dimension.

For the latent interaction layers, we use each context frame vector and the action vector for the correponding timestep t for the context frame feature regularization, we use the aggregated average context frame feature $z_{x_{\bar{t}}}$ to form the context vector for the current action features.

For the experiments comparing to unimodal action sensories, we use our own method for encoding these modalities and conditioning video model. For the sensory modalities of muscle EMG and hand forces, there lacks research works concerning the senses of muscle activation and haptic forces. For hand poses, most works concerning hand poses tackle the task of detection of hand regions from videos [34, 51, 73]. Therefore they also cannot be directly adapted to compare with our work. For this reason, we use our own method for encoding these modalities and conditioning video model.

For experiments on down stream application, we follow the original diffusion policy implementation. The image prompted DP (Sec. 4) uses ResNet [25]-18 image encoder, and the text prompted DP (Sec. 7.8.10) uses OpenClip [1] text-encoder. We modify the original 1D UNet to be four layers with hidden dimensions set to [128, 256, 512, 1024]. The dimension of action space comes to 2292, with two hand poses $24 \times 3 \times 2$, one body pose $28 \times 3 \times 1$, two arm muscle emg 8×2, two hand forces is $32 \times 32 \times 2$.

Hardware, Software, Training Setup We use a server with 8 NVIDIA H100 GPU, 127 core CPU, and 1T RAM to train our models for 15 days. We implement all models using the Pytorch [50] library of version 2.2.1 with CUDA

time frame	t
history horizon	$[0, t-1]$
future frames	$[t-1, T]$
video frame	x_t
encoded video frame	z_{x_t}
action modality	m
action modality signal	$a_{t,m}$
encoded action modality m signal at time step t	$z_{t,m}$
j-th dimension of encoded action modality m signal at time step t	$z_{t,m,j}$
cross-modal feature	y_t
regularized cross-modal feature	y'_t

Table 4. Notation Chart

12.1, and accelerator [20] and EMA [31] . We train our models with batch size of 18 per GPU. We use the Adam [32] optimizer with learning rate of $1e-4$ and betas $(0.9, 0.99)$, ema decay at 0.995 every 10 iterations.

Experimental Setup The ActionSense [13] dataset does not contain the detailed text description used in Sec. 3.1. We generate these text descriptions by using several metrics. We augment the original dataset by resampling video frames, three-ways, every frame, every other frame, and every three frames. We add description of `slow in speed` to the first chunk of data, and `fast in speed` to the third chuck of data. Additionally we also calculate the average hand force magnitude for every task. If the hand force sequence contains frames that are significantly larger than the average frame we add `holding tightly` and add `holding gently` to the lowest force data sequences.

7.5. Additional Pipeline Figure

W provid additional pipeline Fig. 11.

7.6. Model Size

We report the modules of our model in Table. 5. We can see that the multimodal action signal module is fairly small compared to the video module. Each signal average to around 18044828 parameters which is only 5 percent of the total model weights. The lightweight action signal heads highlights the advantage of our method for low computational cost added for each action signal modality

module	parameter count	percentage of total
signal expert encoder	43780932	0.13
signal projection	11537408	0.03
signal decoder	28398382	0.08
signal Total	83716722	0.25
video model	252380168	0.75
total model	336096890	1.00

Table 5. Parameter Count on 64×64 model.

Additionally, people are frequently concerned the real-time execution and edge device computing. We would like to highlight that our work proposes a multisensory conditioned video simulator. When employed in robotics applications, simulator are used in to train policy networks. Normally, only the trained policy network, rather than the simulator itself, needs to be deployed on edge devices / robots. In general, simulators, including ours, do not require to be executed on edge devices or robots for real-time deployment.

We show such application in Sec. 4 Downstram application. Similar to UniSim or any other robotic simulators, we train a goal-conditioned policy network using our pretrained video model. We directly adopt diffusion policy [9] as our policy network, which is lightweight (shown below) and can be executed on Jetsons as shown below, the parameter count for the policy network trained in Table. 6.

7.7. Discussion of Limitations and Future Work

Our experiments are conducted on datasets of human actuation and activities. Ideally, it would be interesting to see the deployment of planned and optimized policies on real humanoid robots with similar multi-sensory capabilities. Because we currently do not have such hardware setup that enables dense force readings on human-hand-like robotic hands or various other fine-grained interoceptive modalities on humanoid robots. We leave this direction for a future research.

There are other passive exteroceptive senses that can be combined with vision, such as depth, 3D and audio etc. One can directly leverage a multi-branch visual-audio or visual-depth UNet diffusion model as the backbone to achieve such multi-modal experoception responses. However, due to limited availability of such data, we leave this direction as future work.

Additionally, because of limited computational resources, we limit our video diffusion model to be very low resolution. However, one can employ upsampling approaches to map low-resolution video predictions to higher resolution. Our work is less concerned with the specifics of image quality but more with the application of using multi-sensory interoception data. Therefore, we leave the study of low-cost video upsampling or better video diffusion backbone as future work.

7.8. Additional Experiments and Discussion

7.8.1. Text as addition to multisensory actions

We are also interested in learning whether multi-sensory action can entirely replace text as condition. We integrate an additional text-encoder head to the MoE feature encoding branches to incorporate simple text phrases, *e.g.* `cut potato`. The encoded text features are aggregated with other multi-sensory action features in the same manner as described in Sec. 2.1. We use the pretrained OpenClip [28] text encoder to encode text in all baselines and our model.

As depicted in the bottom half of Figure. 7, when multiple objects (pan and plate) appear in context image and when the action trajectory can be applied to both objects, the network is uncertain about which object to apply the action. It cleans the plate instead of the pan. When we add text description `clean pan` as an extra piece of information, ambiguity is removed and accurate video can be generated. We also observe that when the context frame is not ambiguous, multi-sensory action provides enough information to generate accurate video trajectories. Adding additional text feature induces a temporal smoothing effect generating similar images across frames.

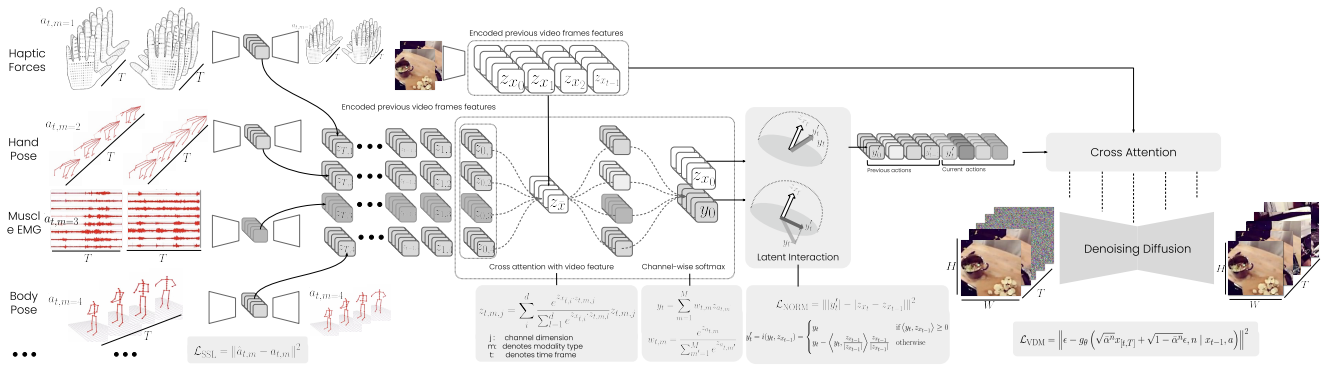

Figure 11. Additional pipeline figure.

module	parameter count	float16 in MB	float32 in MB
policy network (to be deployed on edge devices)	120690484	241MB	482 MB

Table 6. Parameter Count for the policy network model used in Downstream application section.

Hardware Type	NVIDIA Jetson Nano	Jetson Xavier	Jetson Orin NX	Jetson AGX Orin	RTX 4090	H100
Throughput (FPS)	166 ∼ 111	415 ∼ 290	1,725 ∼ 1,293	2,555 ∼ 1,916	26,528 ∼ 19,896	315,141
Latency (ms)	6.6 ∼ 9.2	2.4 ∼ 3.44	0.57 ∼ 0.77	0.39 ∼ 0.52	0.037 ∼ 0.050	0.00317
Energy Cost(J)	0.06 ∼ 0.09	0.036 ∼ 0.051	0.0114 ∼ 0.0154	0.0195 ∼ 0.026	0.01665 ∼ 0.02250	0.02219

Table 7. Table shows that the trained policy can be deployed onto Edge devices.

Method	MSE ↓	PSNR ↑	LPIPS ↓	FVD ↓
No hand pose	0.138	14.1	0.314	264.0
No hand force	0.129	14.5	0.317	256.3
No body pose	0.137	14.5	0.322	273.1
No muscle EMG	0.121	15.2	0.311	217.1
All sensory used	0.110	16.0	0.276	203.5

Table 8. Training with ablated modalities

Table 9. Testing with single modality available

Method	MSE ↓	PSNR ↑	LPIPS ↓	FVD ↓
Hand pose	0.121	14.6	0.309	210.2
Hand force	0.117	14.7	0.307	208.0
Body pose	0.123	14.6	0.310	210.5
Muscle EMG	0.132	13.9	0.312	214.8
All sensory used	0.110	16.0	0.276	203.5

7.8.2. Additional results on training with missing modalities

We first ablate different sensory signal input, when training our video simulator. We observe that body pose is crucial for larger motions that involve moving in space such as turning or walking. For more delicate manipulations such as cutting or peeling, hand poses and haptic forces get us most of the way. Results in Table 8 suggests that contribution of muscle EMG is minimal. A closer look into the dataset reveals that muscle EMG is highly correlated with hand force magnitude, but it provides extra information in scenarios where hands are fully engaged.

7.8.3. Additional results on test-time robustness

As we see from the Table. 9 that when one modality is provided, our model can still produce higher prediction accuracy compared to text-based models or single-model models. Comparing this result with Table. 1 shows that our proposed multsensory action trainiing strategy induces higher quality action feature compared to training with a single modality. This comparison indicates that through implicit association between different modalities, both feature alignment and information presevation is achieved. That is, the complementary information is preserved in the feature representation such that when only one action modality is provided, the model might have access to commonly co-activated feature dimensions and thus produce better result than training with single modality.

To provide a comprehesive set of ablation studies on testing with missing modalities, we show Table 10 that includes all possible pairs of modalities used during testing. The results in Table. 10 along with Table. 9 and Table. 3a makes a

comprehensive study cross all possible ablated experiments. We can from Table.10, that the model achieves better performance when different aspect of information is provided.

Table 10. Testing with paired modality available

Method	MSE ↓	PSNR ↑	LPIPS ↓	FVD ↓
Hand Pose and Hand Force	0.115	14.9	0.304	206.4
Body Pose and Muscle EMG	0.122	14.6	0.309	210.1
Hand Force and Muscle EMG	0.117	14.7	0.307	207.6
Hand Pose and Body Pose	0.113	15.0	0.297	206.2
All sensory used	0.110	16.0	0.276	203.5

7.8.4. Comparison between Training and Testing with Ablated Modalities

The critical difference between the above two experiments, training with ablated modalities (Table. 8) and testing with missing modalities (Table. 3a) is the modalities used during training. The latter ablation experiment, testing with missing modalities, employs a model trained with all modalities, whereas the former is trained only on a subset of modalities. Comparing the performance decrease in Table. 8 and Table. 3a, we can see that the latter experiment, testing with missing modalities, induces very minimal drop in prediction accuracy. This comparison confirms the advantage of training on multimodal action signals. We believe that this test-time robustness is induced by channel-wise attention and channel-wise softmax module, as these design choices allows the model to leverage substitutional information in the given modalities to bridge different modalities to allow for robustness during inference.

7.8.5. History Horizon.

Finally, we study the effect of history horizon length on our model with comparison to text-conditioned simulation. We follow prior works [71] to compare context frame length h(x)=4 and h(x)=1, shown in Table 11. We can see that increased history frame length reduces prediction error for all methods. Additionally, our proposed multisensory action condition is temporally fine-grained, which allows the cross attention between action and observation history h(x, a) = 4 to help further increase simulation accuracy.

7.8.6. Cross Subject Testing

We report the cross subject testing, where we use three other different subjects for testing and training with the rest using the ActionSense dataset, result can be found in Table. 12.

7.8.7. Examples of fine-grained control

We can see from Fig. 12 where hand force together with hand pose helps accurately controls the timing of the hand grabbing the pan.

Method	MSE ↓	PSNR ↑	LPIPS ↓	FVD ↓
Unisim h(x) = 1	0.177	12.7	0.408	674.9
Unisim h(x) = 4	0.118	14.6	0.321	275.9
Ours h(x) = 1	0.142	12.9	0.362	535.1
Ours h(x, a) = 1	0.138	12.7	0.356	529.1
Ours h(x) = 4	0.114	15.4	0.306	256.3
Ours h(x, a_h) = 4	**0.110**	**16.0**	**0.276**	**203.5**

Table 11. Effects of history horizon length

Table 12. Cross Subject Testing

Method	MSE ↓	PSNR ↑	LPIPS ↓	FVD ↓
subject 2	0.115	15.8	0.301	206.7
subject 4	0.112	16.0	0.282	204.6
subject 5	0.110	16.0	0.276	203.5

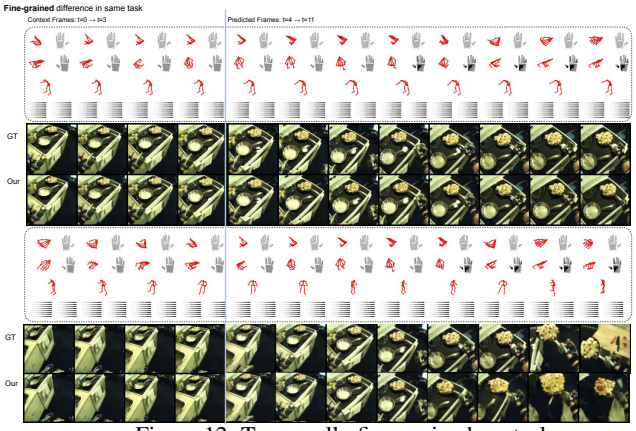

Figure 12. Temporally fine-grained control

7.8.8. Additional Experiment on Generalization to OOD data through Finetuning

We present a second experiment to demonstrate that our method can handle specific out-of-distribution (OOD) scenarios through fine-tuning. For this experiment, we modified the original ActionSense dataset to create OOD data. Using LangSAM, we extracted segmentation masks for "potatoes" and recolored them to appear as "tomatoes." Since the video model had not encountered red vegetables or fruits during training, we fine-tuned our pretrained model on a small dataset of approximately 600 frames (30 seconds) and evaluated it on the test split of this "tomato" data. The data creation procedure is shown in Fig. 13 and results on this experiment can also be found in 14. The results show that the model achieves reasonable performance after fine-tuning. While we acknowledge that robust in-the-wild generalization requires training on larger-scale datasets with diverse domain coverage, this experiment illustrates a practical use

case for addressing OOD data. Specifically, it demonstrates that by collecting a small, specialized dataset, our pretrained model can be effectively fine-tuned to adapt to new domains.

7.8.9. Additional discussion and results on downstream application

Sample results visualization can be found in Fig. 15. We also observe from the figure that the policy optimized by our proposed approach can be different from the ground truth action trajectory, yet the simulated visual observations still closely resemble the ground truth state observations. We believe that the softmax aggregation learns to pick out information deemed useful by the simulator, leaving freedom in irrelevant dimensions in the action space.

7.8.10. Downstream Application2: Multi-Sensory Action Planning

Another potential downstream application is long-term planning. Inspired by [17], we use text to describe high-level goals to generate a set of executable next-step actions. Our video model takes an image observation and the generated actions to simulate future image sequences, which can be further evaluated for next-step execution planning. As shown in Fig. 15, our model can potentially be used for low-level actuation planning through iterative action roll outs. We adapt diffusion policy (DP) [9] to take in both first frame image feature x_0 and high-level goal γ described by a text feature f_γ as the context conditions to generate multi-sensory trajectories of fine-grained actions $a_{[1,T]} = p(x_0, f_\gamma)$. The action steps are then fed into our action-conditioned video generative model $g(\cdot)$ to generate sequences of future video frames $\hat{x}_{[1,t]} = g(x_0, a_{[1,t]})$. To decide whether the subtask τ has been achieved, we use a vision language model $f_v(\cdot)$ as a heuristic function [48], which can be promted with the end state of the current roll out $\hat{x}_t$ to evaluate whether subgoal τ has been achieved. If more steps are needed, we can further iterate the process $a_{[t,it]} = p(\hat{x}_t, \gamma)$, $x_{[t,it]} = g(\hat{x}_t, a_{[t,it]})$. A sample result from text-promted diffusion policy is shown in Figure. 15. We observe long iterations result in accumulative error, as shown in the bottom row of Fig. 19 in Appendix Sec. 7.8). A larger-scale dataset can further boost performance for this task. This downstream application hints at fully automated low-level motion planning and dexterous manipulation, enabling realization of household robots.

7.8.11. Higher Resolution Results

We include some sample results for higher resolution model of video size $128 \times 128 \times 12$ and $192 \times 192 \times 12$, matching the video resolution of existing generative video simulation paper, such as Unisim [71]. The results are shown in Fig. 18

7.8.12. Additioanl qualitative results on other dataset

To show that our proposed method is generic is not designed for the ActionSense [13] dataset, we conducted an experiment by directly applying our proposed approach on another dataset, H2O dataset [34]. H2O [34] dataset is a unimodal action-video dataset that includes paired video and hand pose sequences. We would love to expand our our training on larger and more diverse dataset, However, to the best of our knowledge, ActionSense [13] is the only dataset that includes paired multisensory action signal monitoring sequences alongside video sequences. We show experiment on H2O [34] in Figure 16. We provide additional sample test on the holoAssist dataset 17, which is also a hand-pose video dataset in Fig. 16. These results demonstrate that our system is generic, not dataset specific, and can achieve reasonable performance. These results indicate that our model is capable of training and testing on unimodal action datasets, highlighting its generalizability beyond the ActionSense dataset. This demonstrates that our method is not specifically tailored to ActionSense and can adapt to various scenarios. We believe our proposed method offers a generalizable framework that can serve as a reference and can be applied more broadly as additional datasets of this nature become available.

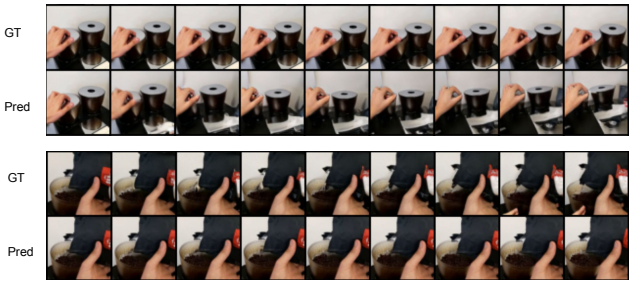

Figure 17. Test on HoloAssist dataset

7.9. Additional Qualitative Results

Additional Qualitative Results are shown in Fig. 19, Fig. 20, and Fig. 21. Fig. 19 and Fig. 20 show additional qualitative results of context frames and predicted video frames from our proposed multisensory action signals. Fig. 21 shows demonstrations of failure cases, policy optimization, and long-trajectory planning. We show one most recent context frame and the eight predicition frames. Fig. 21 shows results paired in two rows, where the top row shows ground truth trajectory the bottom row shows predicted trajectory.

7.9.1. Failure Cases

We show the failure cases on the top right section. Common failure cases include false hallucination of environment with large motion. Failure to identify object with similar apperance to background. The wooden chopboard gradually disppear into the wooden table background and fails to pick it up in simulation. Failure in identify object to act on (also hallucates pan handle on plate and cleaning the plate). The last five rows in Fig. 19 show additional results on down stream tasks of policy planning, shown in the middle rows, and long-trajectory simulation, show in the bottom row.

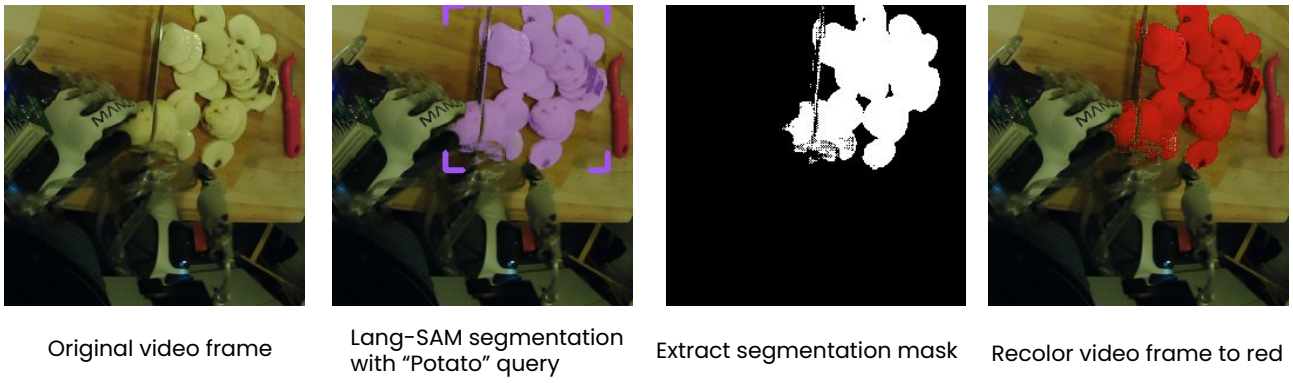

Figure 13. Experimental set up on OOD testing.

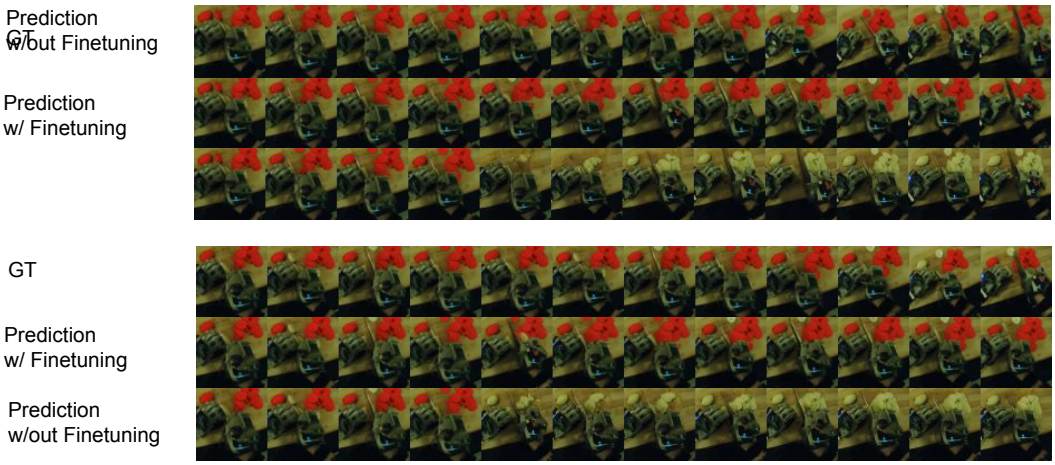

Figure 14. Experimental results on OOD testing.

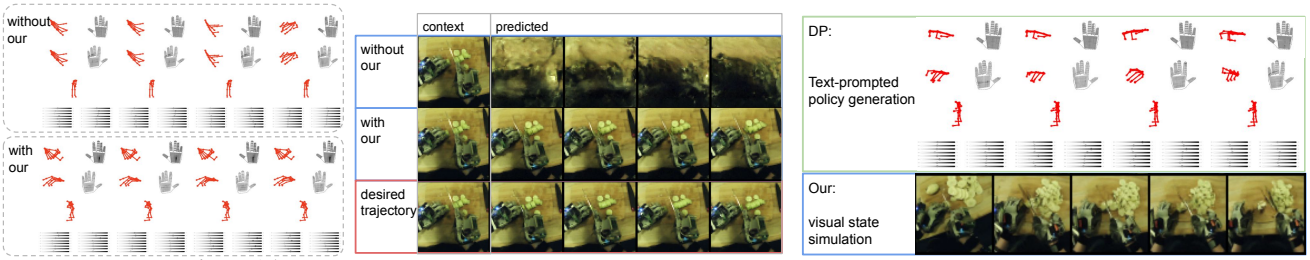

Figure 15. **Left:** Results on goal-conditioned policy optimization. **Right:** Results on long-term task planning.

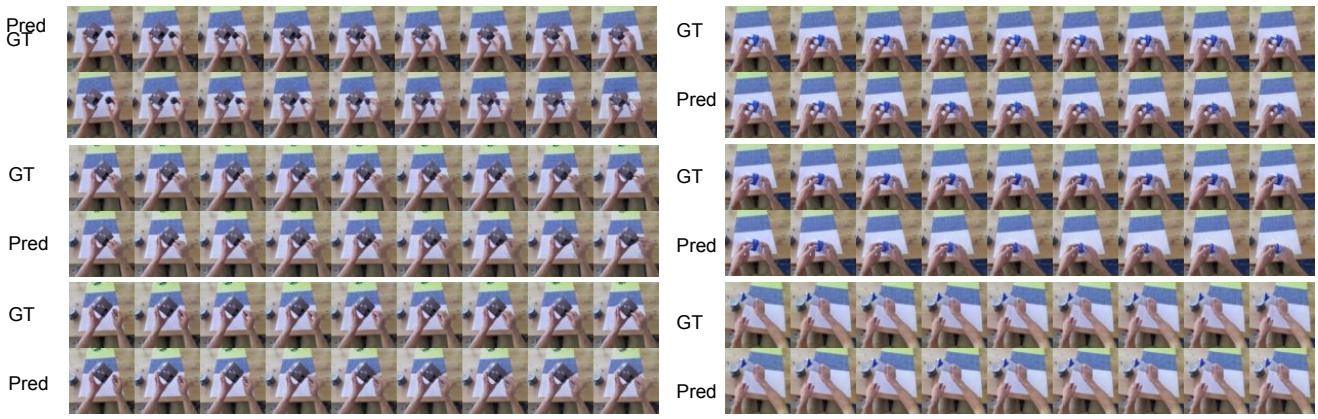

Figure 16. Test on H2O dataset

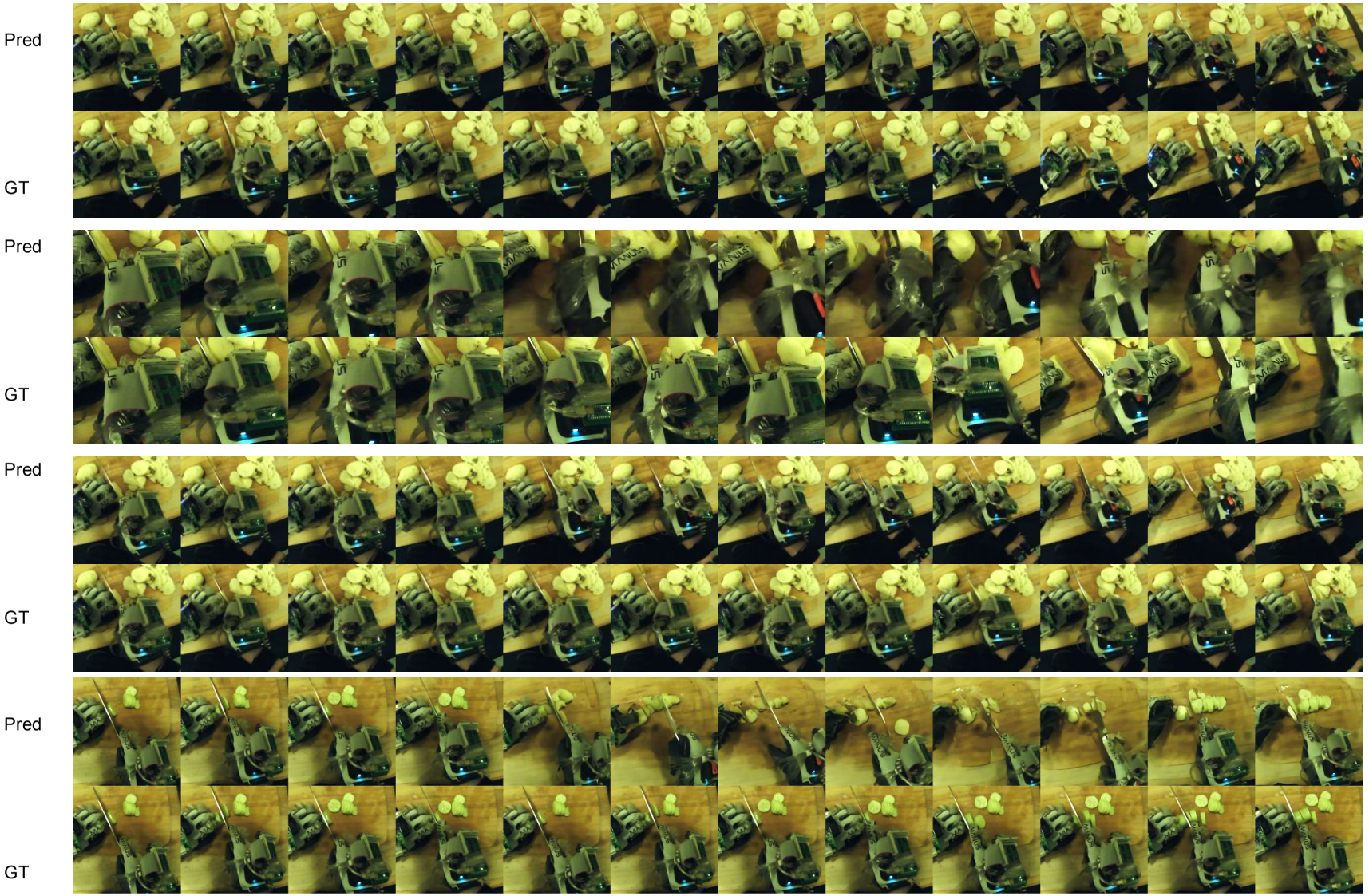

Figure 18. The left three are of resolution 128 and the last one is of resolution 192

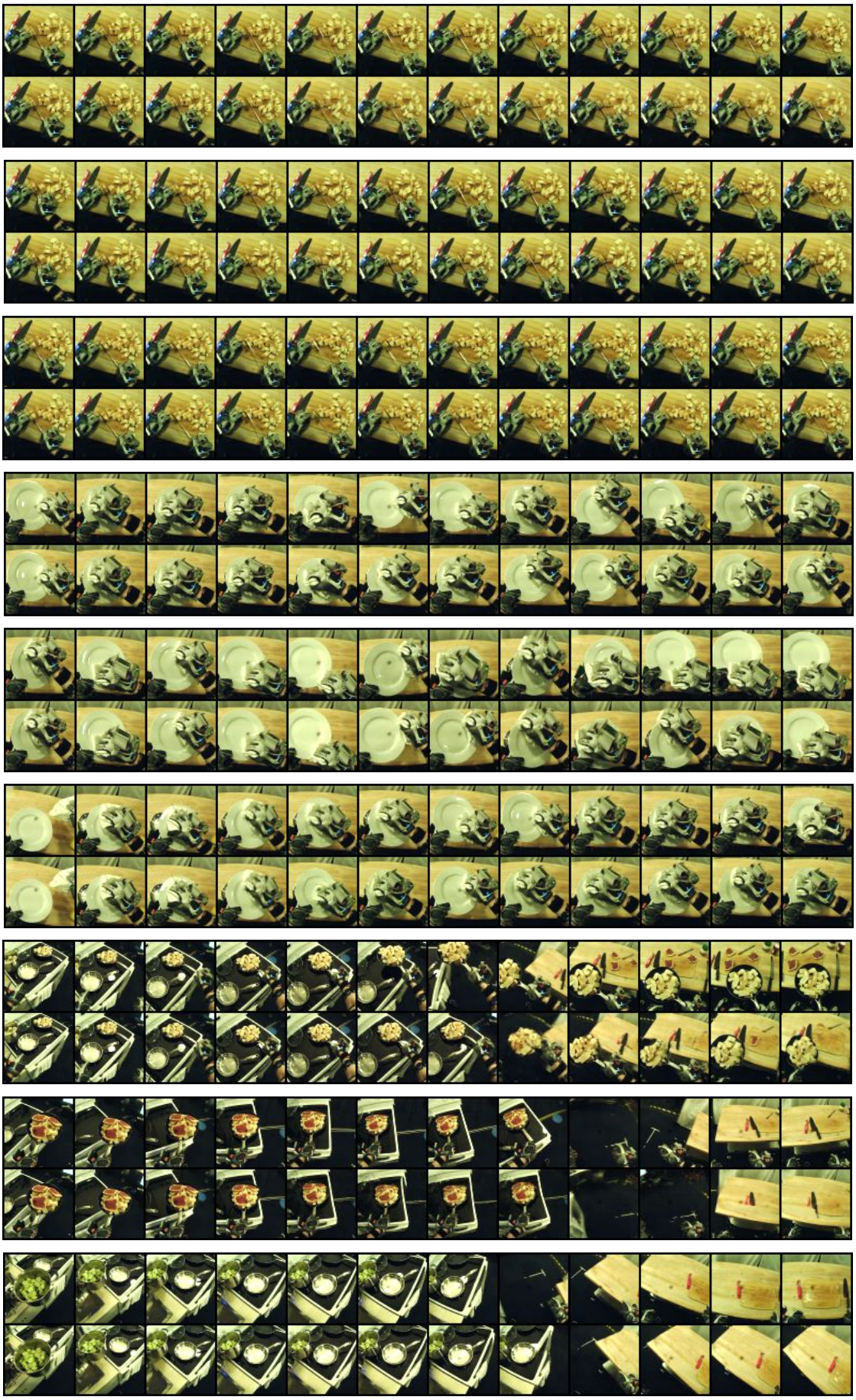

Figure 19. Additional qualitative results

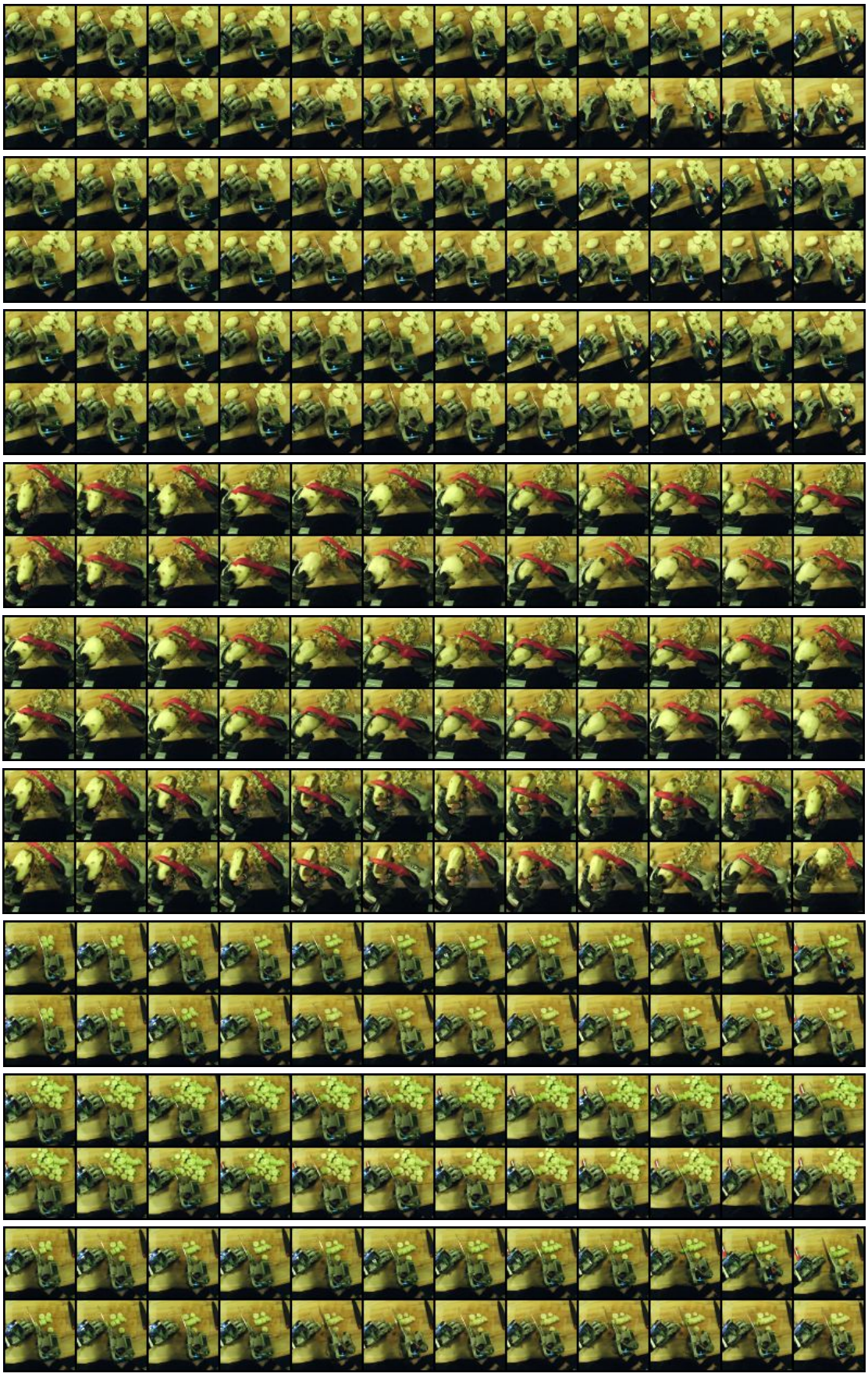

Figure 20. Additional qualitative results

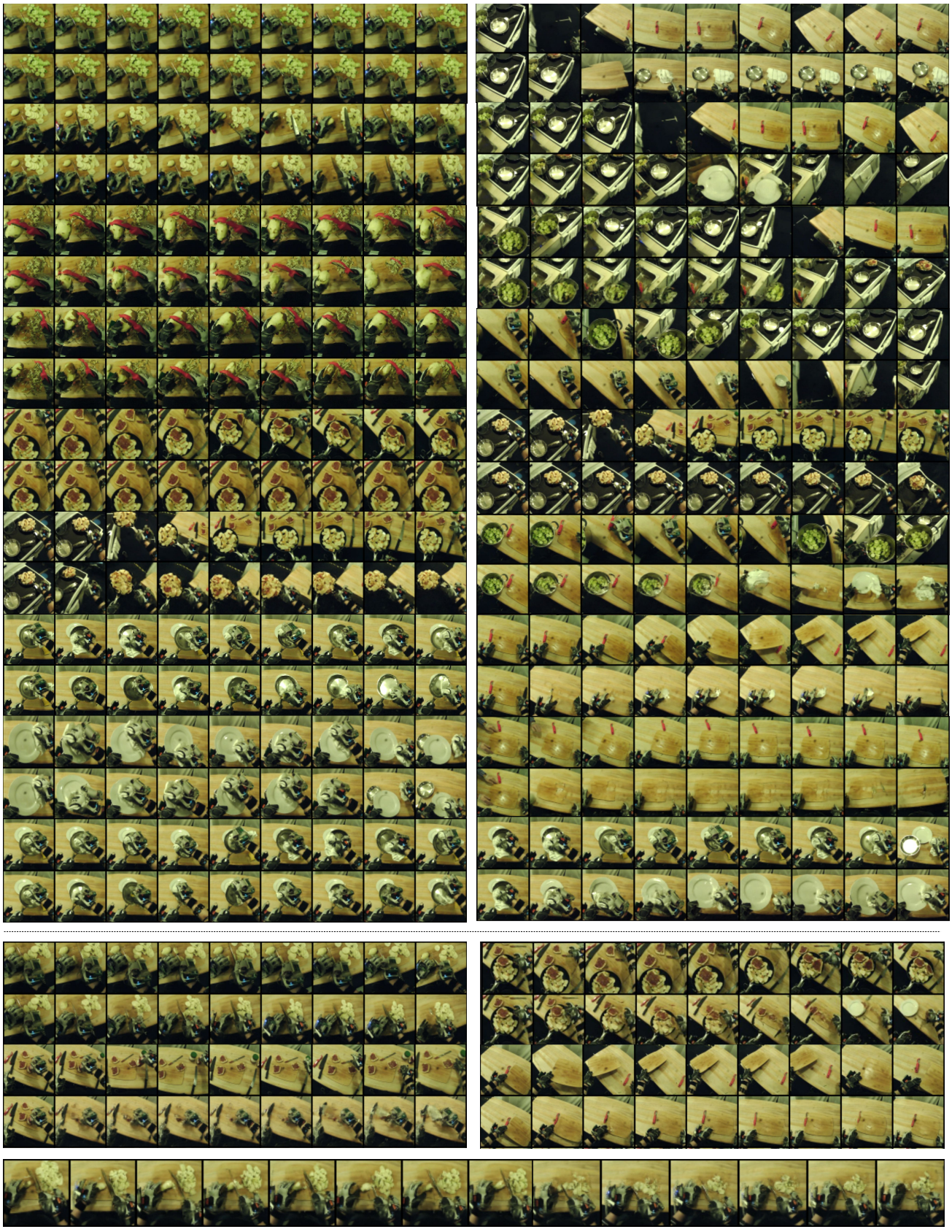

Figure 21. **Top left:** Additional qualitative results. **Top right:** Failuare cases. **Middle left and right:** Additional results on policy optimization. **Bottom:** long-trajectory policy planning.