

A Model Details

A.1 Encoder

As described in Section 3 of the manuscript, we use a ResNet-18 encoder [A7] to extract image features, producing a feature map of shape (F, H, W) with $F = 512$. For Meta-World [A10] images at a resolution of $(224, 224)$, this results in $(H, W) = (7, 7)$. For other benchmarks with input resolution $(240, 320)$, the feature map has dimensions $(H, W) = (8, 10)$. Following prior work [A3, A5, A2], we initialize the ResNet-18 with ImageNet-pretrained weights to facilitate better convergence.

A.2 Action and World Models

At each timestep i , the expert states e_N, \dots, e_1 and the agent states r_i, \dots, r_1 are concatenated along the temporal dimension, resulting in a tensor of shape $(N + i, F \times H \times W)$. Sinusoidal positional embeddings are then added along the time dimension, and the sequence is processed by causal transformer blocks to produce the latent action a_i . For Meta-World [A10], we use 6 causal transformer blocks, whereas for other benchmarks, we use 2 blocks. All transformer blocks use 4 attention heads. For the world model, the latent actions a_i, \dots, a_1 are concatenated with the corresponding states along the feature dimension and passed through a single causal transformer block with 4 attention heads. The output at the final timestep is projected to shape $(F \times H \times W)$ to predict next state r_{i+1} .

A.3 Grasp Correction

Following the approach in [A2], we use a depth camera mounted on the robot’s end-effector to refine the grasp pose. Specifically, when the agent reaches a waypoint that requires closing the gripper (i.e., initiating a grasp), it uses the depth camera to detect all visible connected components and identifies the largest one. The end-effector is then adjusted so that this largest contour is positioned directly beneath it. In practice, this grasp correction step is repeated for three consecutive iterations each time a grasp is triggered.

B Training Details

As mentioned in the manuscript, we train the model for 10 epochs on simulation benchmarks and 100 epochs on real-world benchmarks. Training is performed using the AdamW optimizer with a one-cycle learning rate scheduler and a batch size of 128. All models are implemented in PyTorch [A8] and trained using the accelerate library [A6] for multi-GPU parallelization across 4 NVIDIA A100 GPUs (40 GB each). For evaluation, simulation benchmarks are run on an RTX A4000 GPU with 128 GB RAM and an Intel i9 CPU. Real-world benchmarks are evaluated on an RTX 2080 Ti GPU with 64 GB RAM and an Intel i7 CPU.

To improve generalization, we adopt the demonstration image mixup strategy, similar to prior work [A2]. During training, each input image \mathcal{I} is linearly mixed with an image \mathcal{J} from another task using a mixing factor $\beta \sim \mathcal{U}(0.3, 1.0)$ where \mathcal{U} is the uniform distribution. The resulting input is $\tilde{\mathcal{I}} = \beta\mathcal{I} + (1 - \beta)\mathcal{J}$. This encourages the model to rely on task-relevant cues rather than image semantics. Additionally, we apply regularization techniques such as random cropping, random translation, and color jittering to enhance robustness and performance.

Loss Functions: As described in the manuscript, we use the L1 loss for supervising the world model’s outputs and the differentiable Soft Dynamic Time Warping (Soft-DTW) [A4] loss for training the predicted waypoints. Dynamic Time Warping (DTW) is commonly used to measure the similarity between temporal sequences that may vary in length or speed, by solving a minimum-cost alignment between them using dynamic programming. However, the standard DTW loss is not differentiable with respect to its inputs, which limits its applicability in gradient-based optimization. To address this, [A4] introduced Soft-DTW, a differentiable relaxation of DTW that replaces the hard minimum operator in the dynamic programming recurrence with a soft-minimum (log-sum-exp) formulation. This soft assignment aggregates alignment costs across all possible paths, weighted by their relative scores, thereby allowing gradients to flow through all potential alignments. In our work, we adopt this

differentiable Soft-DTW loss to effectively supervise the predicted waypoint sequences and enable end-to-end training.

C Benchmarks Environments

C.1 Meta-World

The Meta-World benchmark [A10] comprises 50 diverse manipulation tasks performed using a Sawyer robotic arm. Following the setup in [A2], we use 46 tasks for training and reserve 4 tasks for testing. For each of the 46 training tasks, we collect 100 trajectories from both the expert and the agent across varied, randomly sampled environment configurations. These trajectories are used for training the model. The test tasks are grouped into ‘easy’ and ‘hard’ categories.

The easy tasks include Pick-Place-Wall-V2 and Button-Press-V2. Pick-Place-Wall-V2 differs from Pick-Place-V2 from the training set by introducing a wall between the pick and place locations. As a result, the manipulator must lift the object higher to avoid collision, posing a non-trivial challenge. Similarly, Button-Press-V2 differs from Button-Press-Wall-V2 in the training task by omitting the wall, altering the spatial context.

The hard tasks are Window-Open-V2 and Door-Unlock-V2. Window-Open-V2 is semantically similar to Window-Close-V2 from the training set but requires performing the reverse action. Successfully completing this task requires the model to follow the expert demonstration rather than relying solely on scene semantics. Likewise, Door-Unlock-V2 is the reverse of the training task Door-Lock-V2, also demanding fine-grained control and precision, making it particularly challenging.

C.2 Pick-and-Place

The Pick-and-Place benchmark, introduced in T-OSVI [A5], features a tabletop environment with four distinct objects, varying in size and shape, placed on one side, and four target locations on the opposite side. This setup defines 16 unique pick-and-place pairs, each representing a separate task within the benchmark. Following [A2], we use 15 tasks for training and reserve 1 task for testing.

A key challenge in this benchmark is the embodiment mismatch, where the expert demonstrations are performed using a Sawyer robotic arm, while the agent executes tasks using a Franka arm. As in the Meta-World setup, we collect 100 trajectories for each training task from both the expert and the agent under randomly sampled environment configurations, which are used to train the model.

C.3 Two-Franka-PP

This benchmark involves two Franka robotic arms mounted at different locations on a shared tabletop environment, as illustrated in Fig. 5a of the manuscript. The gray Franka serves as the expert, while the white Franka acts as the agent. We utilize an Intel RealSense D455 camera with known intrinsic and extrinsic (camera-to-robot transformation) parameters, capturing images at 640x480 resolution. Additionally, an Intel RealSense D435 is mounted on the robot’s end-effector to assist with grasping, as detailed in Section 3.2 of the manuscript. We use these same set of cameras for Human-Franka-PP and Human-Franka-Push benchmarks as well. The task in Two-Franka-PP is a pick-and-place operation involving two objects: a soup can and a sugar box. The corresponding target locations are positioned near the center of the table, marked with green and orange paper, respectively. Due to the high cost of real-world data collection, we sample 30 trajectories, with different configurations of the environment, each for the expert and the agent across four tasks. Following the same train-test split protocol as prior benchmarks, we use 3 tasks for training and 1 for testing. This real-world setup introduces key challenges like domain shift arising from the differing mount positions of the expert and agent arms, and the limited number of trajectories available for training.

C.4 Human-Franka-PP

This benchmark adopts a similar pick-and-place setup as Two-Franka-PP, but replaces the expert with a human demonstrator, while the agent remains a Franka robotic arm, as shown in Fig. 5b of the manuscript. In addition to the change in embodiment, the position and orientation of the external camera, as well as the locations of the target areas, are modified in this setting. As in the

previous real-world benchmark, we collect 30 trajectories per task with different configurations of the environment. We use 3 tasks for training and reserve 1 for testing. The high embodiment gap between the human expert and robotic agent, combined with the limited number of trajectories, makes Human-Franka-PP a particularly challenging benchmark.

C.5 Human-Franka-Push

This benchmark features a tabletop environment with a blue and a green cuboid-shaped objects. The objective is to either push or pull one of the objects, resulting in a total of four distinct tasks. We use 3 tasks for training and 1 for testing. In this setup, the expert is a human demonstrator, while the agent is a Franka robotic arm. As with the other real-world benchmarks, we collect 30 trajectories each from the expert and the agent, using varied configurations of the environment to promote generalization. Beyond the embodiment gap between the human and robotic agent, this benchmark poses an additional challenge: the required manipulations demand highly precise, fine-grained control. The low margin for error in object interactions makes accurate trajectory execution critical for successful task completion.

D Additional Comparisons

D.1 Visualization Examples

Waypoint Comparisons In this section, we compare the waypoints predicted by OSVI-WM (Fig. A1b) and AWDA[A2] (Fig. A1a) on an example from the Human-Franka-Push benchmark. This particular example was selected because it comes from a real-world setting and requires fine-grained control for successful execution. As shown in the figure, the waypoints predicted by AWDA, particularly w_2 and w_3 , cause the robot to collide with the object due to imprecise trajectory planning. In contrast, the waypoints generated by OSVI-WM are more accurate and better aligned with the task requirements, enabling smoother manipulation and resulting in higher success rates.

Agent Rollouts We show visualization examples for each test task of Meta-World [A10] in Figs. A2, A3, A4, and A5, for Pick-and-Place [A5] in Fig. A6, and for the real-world benchmarks in Figs. A7, A8, and A9. Further, we show a failure case in Fig. A10, and the use of replanning in Fig. A11. Details on the rollouts are provided in the respective captions.

D.2 High-Level Comparison of World Models and VLAs for Robot Manipulation

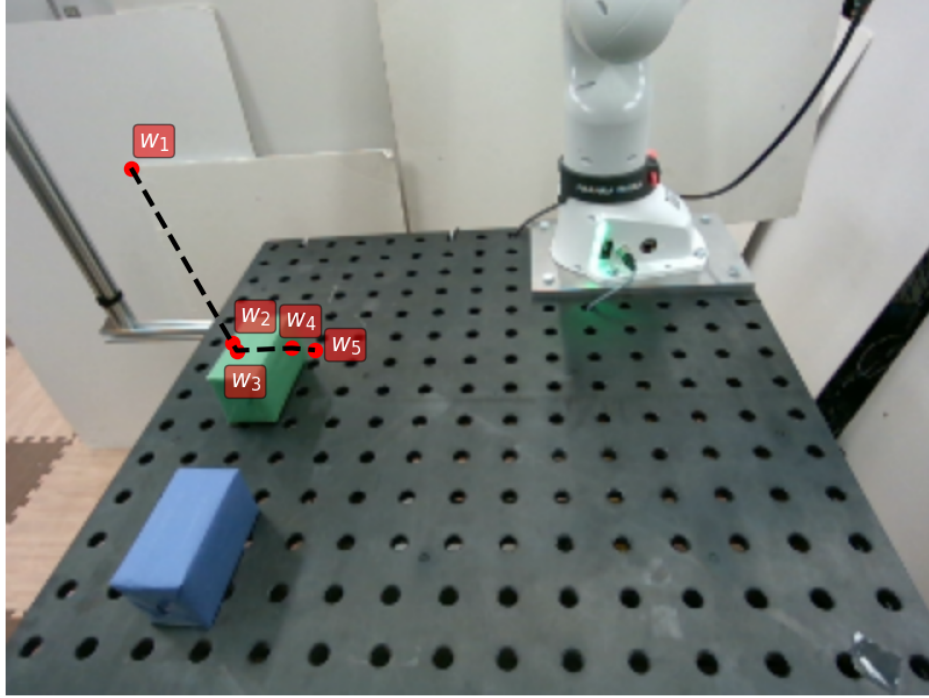
In general, both world-model-based and VLA-based (or behavior cloning) approaches ultimately aim to execute tasks by predicting actions. However, the key distinction is that world-model-based approaches also learn the world dynamics independently of the task. This discourages the model from overly fixating on action prediction and encourages learning more generic features through the additional training signal provided by the world model loss. Most existing methods, whether world-model-based or VLA-based, typically focus on training and testing within the same set of tasks, as detailed in our manuscript. Many of these methods have demonstrated good results in complex dexterous manipulation tasks, as you mentioned. However, when faced with out-of-distribution tasks, like those in our examples, existing BC-based methods such as [A2, A5, A9] often show low success rates, even on top-down grasping tasks. Additionally, our experience indicates that even recent VLA-based robot foundation models like Gr00t [A1] require in-context retraining for new tasks for accurate manipulation, despite being pre-trained on large datasets.

D.3 Efficiency of Re-Planning

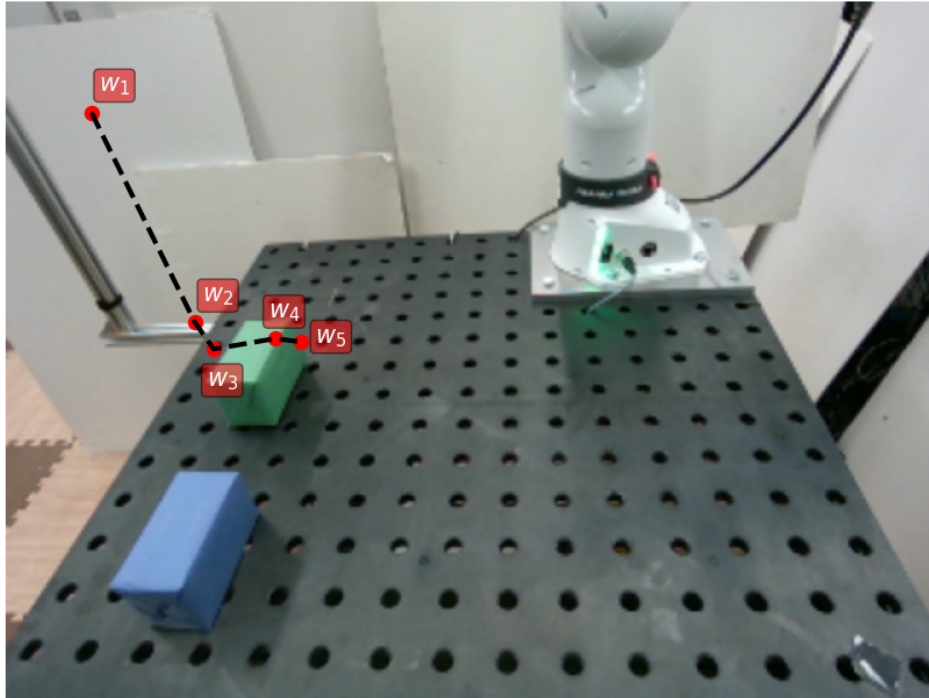
When task execution fails (detected by the simulation environment configuration), OSVI-WM triggers re-planning by taking a new observation and generating a fresh sequence of waypoints based on the robot’s current state. Unlike open-loop retries, replanning is a closed-loop process that takes a new observation of the current robot state and generates fresh waypoints based on that feedback. This closed-loop replanning incurs additional time cost. For the results shown in Figure 8, the time comparison is as follows:

- Without re-planning: Each episode runs for a maximum of 500 steps (40 seconds on our system with one RTX A4000 GPU, 128 GB RAM, and an Intel i9 CPU).
- With re-planning: New waypoints are generated every 500 steps, for up to 2000 steps (160 seconds) total.

Despite this overhead, replanning substantially improves success rates. Notably, in our work, replanning is only applied for the ‘hard’ Meta-World tasks. Even without replanning, OSVI-WM achieves a success rate of 42% on these tasks (71.5% after re-planning), which is already higher than all existing baselines (Fig. 8). Additionally, it is important to highlight that many baseline methods, such as [A5], are also inherently closed-loop but still perform worse than OSVI-WM, further emphasizing the strength of our approach beyond just repeated planning.

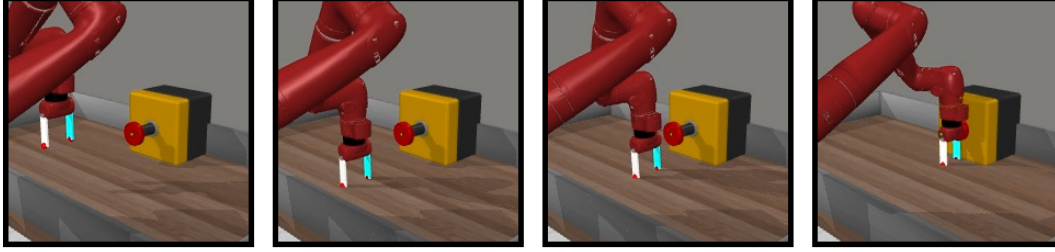


(a) Waypoints predicted by AWDA [A2].

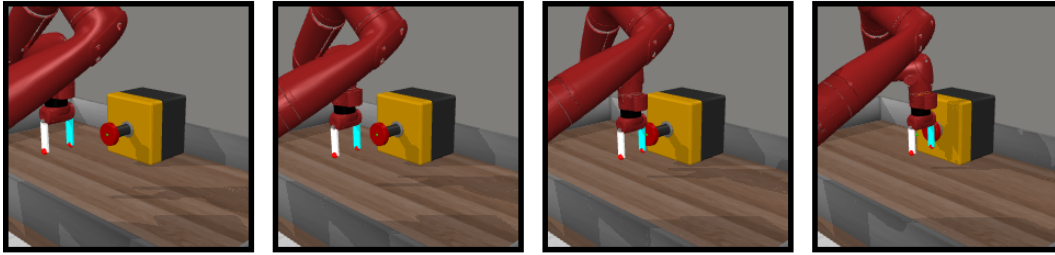


(b) Waypoints predicted by OSVI-WM.

Figure A1: Comparison of predicted waypoints on a Human-Franka-Push example. The waypoints predicted by AWDA, particularly w_2 and w_3 , cause the robot to collide with the object due to imprecise trajectory planning. In contrast, the waypoints generated by OSVI-WM are more accurate and enable smoother manipulation.

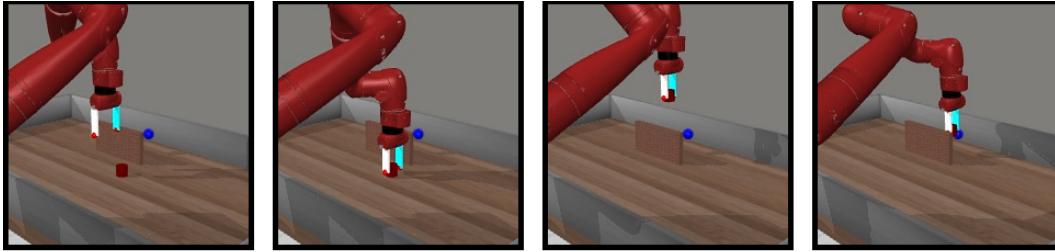


(a) Expert Demonstration

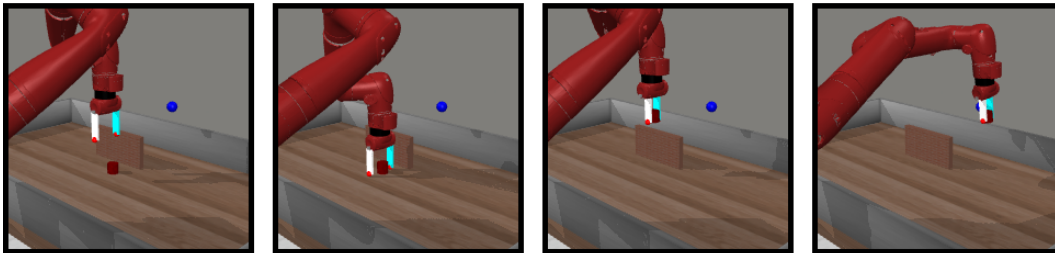


(b) Agent Imitation Using OSVI-WM

Figure A2: **Meta-World [A10] Button-Press-V2**: In this test task, the button is positioned at different locations across the expert and agent runs.

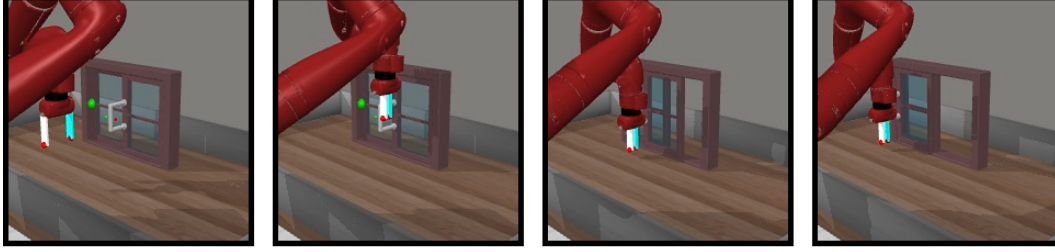


(a) Expert Demonstration

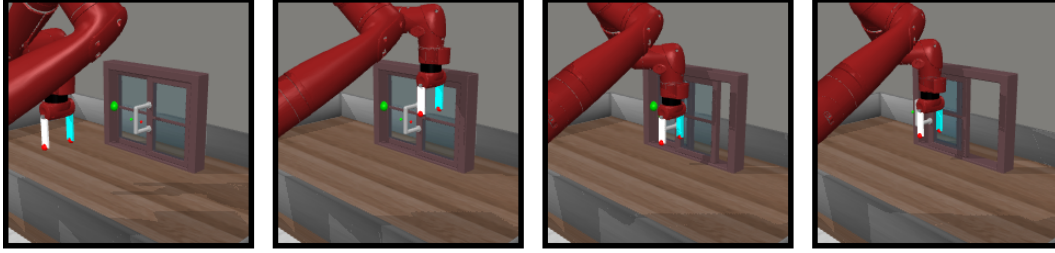


(b) Agent Imitation Using OSVI-WM

Figure A3: **Meta-World [A10] Pick-Place-Wall-V2**: In this test task, the object's initial location, target location (blue dot), and location of the wall are different across the expert and agent runs.

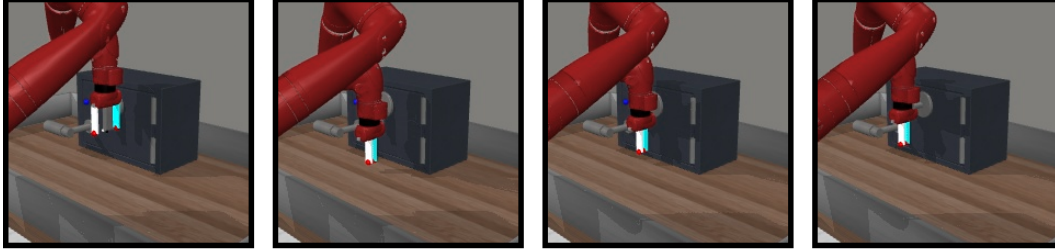


(a) Expert Demonstration

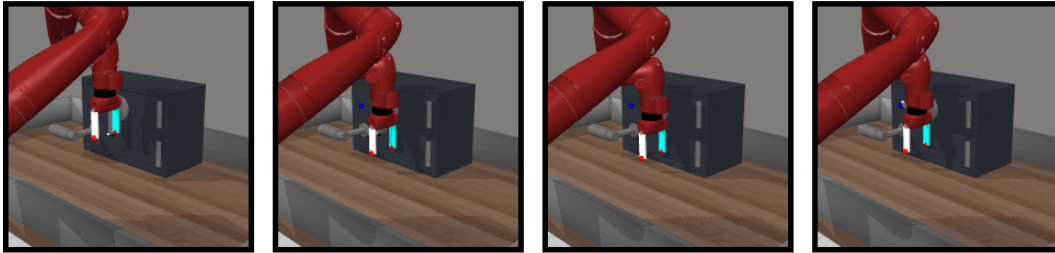


(b) Agent Imitation Using OSVI-WM

Figure A4: **Meta-World [A10] Window-Open-V2**: In this test task, the window is positioned at different locations between the expert and agent runs.

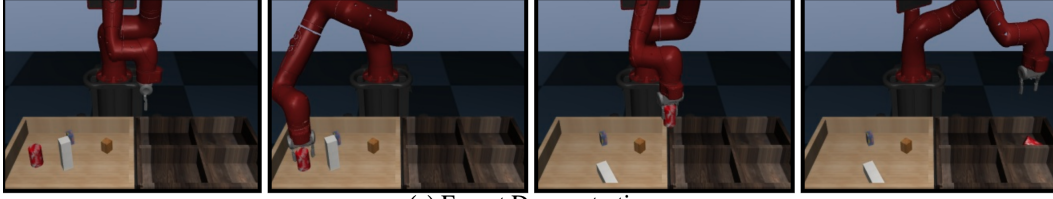


(a) Expert Demonstration

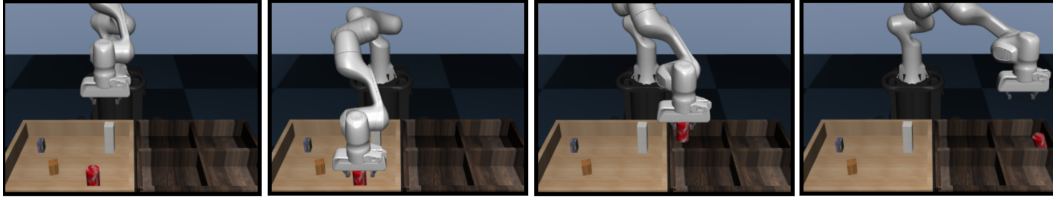


(b) Agent Imitation Using OSVI-WM

Figure A5: **Meta-World [A10] Door-Unlock-V2**: In this test task, the locker (with its door) is positioned at different locations between the expert and agent runs. Further, this task requires finer controls compared to other tasks, making it particularly challenging.

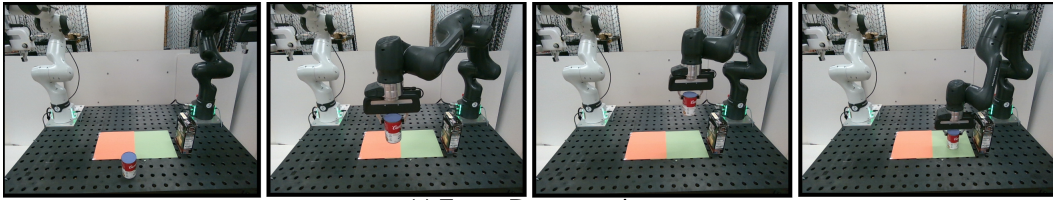


(a) Expert Demonstration

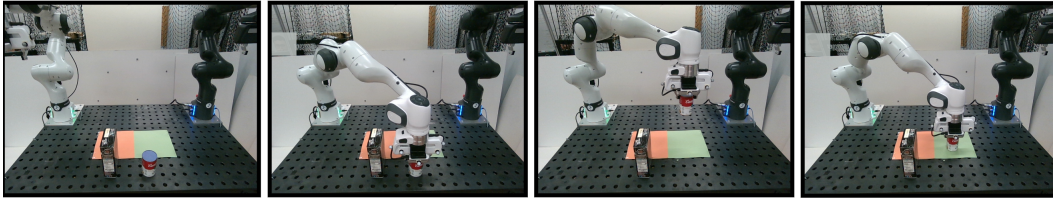


(b) Agent Imitation Using OSVI-WM

Figure A6: **Pick-and-Place [A5]**: The test task involves picking up the red can and placing it in the target location as demonstrated by the expert. The expert uses a Sawyer arm while the agent uses a Franka arm in this setting. Further, the arrangement of the objects is different between the expert demonstration and the agent's imitation rollout.

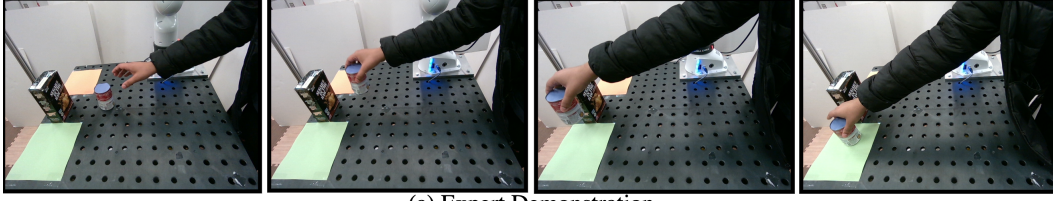


(a) Expert Demonstration

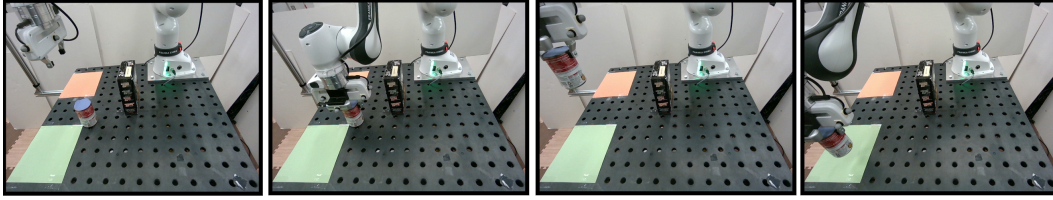


(b) Agent Imitation Using OSVI-WM

Figure A7: **Two-Franka-PP**: The goal is to pick up the can and place it into the green target, as demonstrated by the expert. The expert (gray Franka arm) and the agent (white Franka arm) are mounted at different locations on the tabletop. Additionally, the object arrangements differ between the expert demonstration and the agent's imitation rollout.

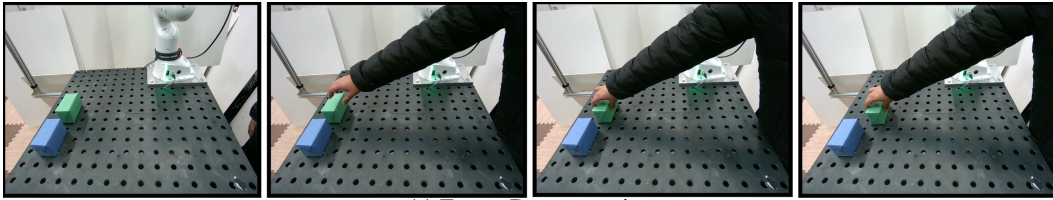


(a) Expert Demonstration

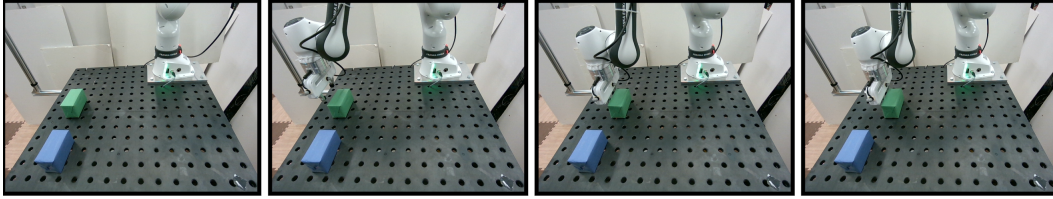


(b) Agent Imitation Using OSVI-WM

Figure A8: **Human-Franka-PP**: The goal is to pick up the can and place it into the green target, as demonstrated by the expert. The expert is a human arm, while the agent is a white Franka arm mounted on a tabletop. Additionally, the object arrangements differ between the expert demonstration and the agent's imitation rollout.



(a) Expert Demonstration



(b) Agent Imitation Using OSVI-WM

Figure A9: **Human-Franka-Push**: The objective is to push the green cuboid towards the agent, as shown in the figure. The expert is a human arm, while the agent is a white Franka arm mounted on a tabletop. Additionally, the object arrangements differ between the expert demonstration and the agent's imitation rollout.

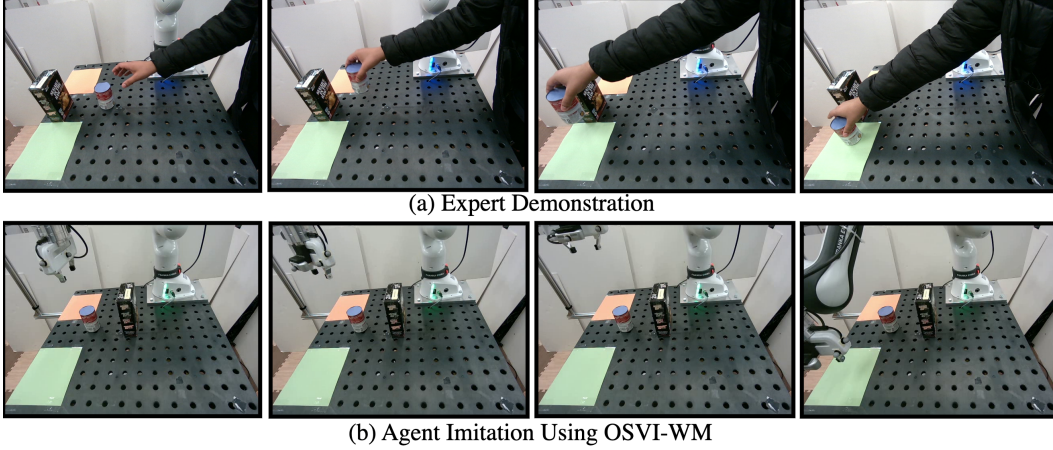


Figure A10: **Failure Case:** This example from the Human-Franka-PP benchmark illustrates a failure case. The agent, aiming to place the can on the green target, initially positions itself above the can but is misaligned by a few centimeters. As a result, it fails to establish a reliable grasp using the end-effector’s depth camera. The agent then skips the grasping stage and proceeds to the next waypoint without the object, ultimately reaching the target empty-handed and failing the task.

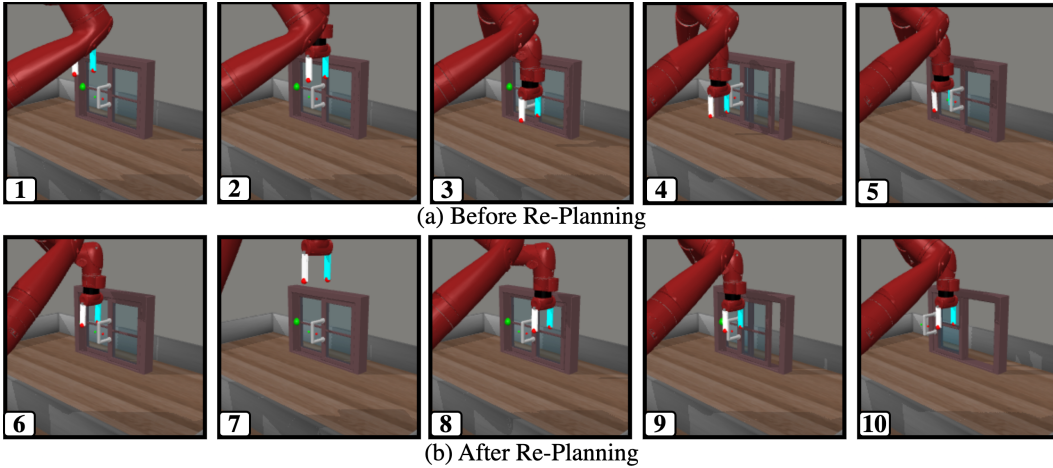


Figure A11: **OSVI-WM’s Re-Planning Example:** The figure shows the agent’s rollout on the Window-Open-V2 test task from Meta-World [A10]. In steps 1–5 (a), the agent fails using the initial waypoints. Re-planning is triggered at step 5, enabling successful task completion in steps 6–10 (b).

References

- [A1] Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.
- [A2] Matthew Chang and Saurabh Gupta. One-shot visual imitation via attributed waypoints and demonstration augmentation. In *Proceedings of the International Conference on Robotics and Automation*, pages 5055–5062. IEEE, 2023.
- [A3] Zichen Jeff Cui, Hengkai Pan, Aadithya Iyer, Siddhant Haldar, and Lerrel Pinto. Dynamo: In-domain dynamics pretraining for visuo-motor control. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Proceedings of the Advances in Neural Information Processing Systems*, volume 37, pages 33933–33961. Curran Associates, Inc., 2024.

- [A4] Marco Cuturi and Mathieu Blondel. Soft-dtw: a differentiable loss function for time-series. In *Proceedings of the International Conference on Machine Learning*, pages 894–903. PMLR, 2017.
- [A5] Sudeep Dasari and Abhinav Gupta. Transformers for one-shot visual imitation. In Jens Kober, Fabio Ramos, and Claire Tomlin, editors, *Proceedings of the Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pages 2071–2084. PMLR, 16–18 Nov 2021.
- [A6] Sylvain Gugger, Lysandre Debut, Thomas Wolf, Philipp Schmid, Zachary Mueller, Sourab Mangrulkar, Marc Sun, and Benjamin Bossan. Accelerate: Training and inference at scale made simple, efficient and adaptable. <https://github.com/huggingface/accelerate>, 2022.
- [A7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.
- [A8] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: an imperative style, high-performance deep learning library. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019.
- [A9] Tianhe Yu, Chelsea Finn, Annie Xie, Sudeep Dasari, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-shot imitation from observing humans via domain-adaptive meta-learning. *arXiv preprint arXiv:1802.01557*, 2018.
- [A10] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Proceedings of the Conference on Robot Learning*, pages 1094–1100. PMLR, 2020.