DyWA: Dynamics-adaptive World Action Model for Generalizable Non-prehensile Manipulation

Supplementary Material

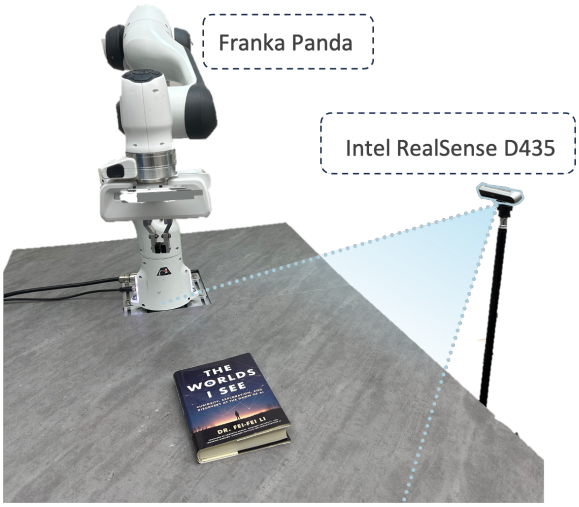

Figure 7. Real-world Experiment Setup

Figure 8. Objects used in the real-world experiments

6. Real-world Setup

As shown in Figure 7, We use a franka panda robot arm for execution and a Intel RealSense D435 camera for capturing point cloud. We also adopt the same object segmentation method with CORN [8], consists of color-based segmentation followed by depth-based back-projection to obtain a single-view object point cloud.

7. RL-based Teacher Policy

7.1. Reward Design

Following Cho et al. [8], Kim et al. [19], the reward function in our domain is defined as:

$$r = r_{suc} + r_{reach} + r_{contact} - c_{energy}, \tag{8}$$

where r_{suc} is the task success reward, r_{reach} is the goal-reaching reward, $r_{contact}$ is the contact-inducing reward, and c_{energy} is the energy penalty.

The task success reward is defined as:

$$r_{suc} = \mathbb{1}_{suc}, \tag{9}$$

where $\mathbb{1}_{suc}$ is an indicator function that returns 1 when the object's pose is within 0.05m and 0.1 radians of the target pose.

To facilitate learning, we introduce dense rewards r_{reach} and $r_{contact}$, formulated based on a potential function [37] as:

$$r = \gamma\phi(s') - \phi(s), \tag{10}$$

where $\gamma \in [0, 1)$ is the discount factor. Specifically,

$$\phi_{reach}(s) = k_g \gamma^{k_d \cdot d_{o,g}(s)}, \tag{11}$$

$$\phi_{contact}(s) = k_r \gamma^{k_d \cdot d_{h,o}(s)}, \tag{12}$$

where $k_g, k_d, k_r \in \mathbb{R}$ are scaling coefficients. The term $d_{o,g}(s)$ represents the distance between the current object pose and the goal pose, measured using a bounding-box-based distance metric, while $d_{h,o}(s)$ denotes the distance between the object's center of mass and the tip of the gripper.

The energy penalty is defined as:

$$c_{energy} = k_e \sum_{i=1}^{7} \tau_i \dot{q}_i, \tag{13}$$

where $k_e \in \mathbb{R}$ is a scaling coefficient, and τ_i and $\dot{q}_i$ denote the torque and velocity of the i^{th} joint, respectively.

7.2. Architecture

World Model	FiLM	Success Rate
✘	✘	94.1
✓	✘	93.9
✓	✓	93.5

Table 5. Success rate of RL-based Teacher Policy.

Our teacher policy architecture consists of separate encoders for each modality and an MLP-based policy network, which proves sufficiently effective for RL training.

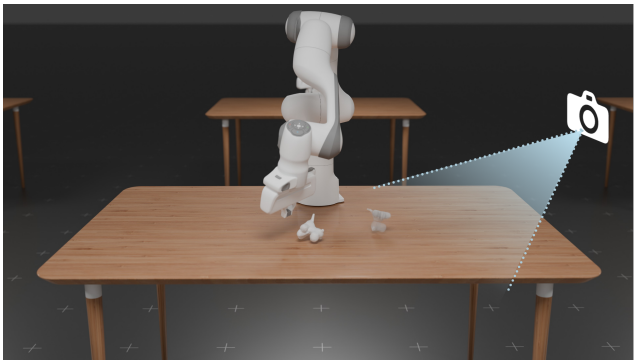

Figure 9. Simulation Environment Setup.

We also experiment with incorporating our proposed world modeling and FiLM into the teacher policy, as shown in Table 5. However, the performance remains nearly unchanged compared to the baseline approach. Together with the analysis in Figure 3, we attribute this to the RL policy already reaching its performance upper bound given the current architecture. The core contribution of DyWA primarily benefits the distillation process, facilitating better optimization of the imitation loss rather than improving the teacher policy itself.

8. Alternative of Dynamics Factor Conditioning

	MLP	Cross Atten	FiLM
Success Rate	73.3	70.1	82.2

Table 6. Success Rate of student policy with different dynamics conditioning methods.

To investigate alternative conditioning mechanisms, we experimented with cross-attention layers as a replacement for FiLM. However, this approach led to significantly degraded performance. We hypothesize that the transformer-based cross-attention mechanism is more sensitive to data distribution shifts and may require additional architectural modifications or extensive data augmentation, introducing unnecessary overhead for this task. These findings further support FiLM as a lightweight yet effective method for integrating adaptation embeddings into the world-action model.

9. Simulation Assets and Setup

Our simulation setup is shown as Figure 9. We sample 323 objects from the DexGraspNet dataset as training set and 10 objects as test set, as shown in Figure 10, 11. To sample stable poses for training, we drop the objects in a simulation and extract the poses after stabilization. In 80% of the trials, we drop the object $0.2\,\mathrm{m}$ above the table in a uniformly random orientation. In the remaining 20%, we drop the objects from their canonical orientations, to increase the chance of landing in standing poses, as they are less likely to occur naturally. If the object remains stationary for 10 consecutive timesteps, and its center of mass projects within the support polygon of the object, then we consider the pose to be stable. We repeat this process to collect at least 128 initial candidates for each object, then keep the unique orientations by pruning equivalent orientations that only differ by a planar rotation about the z-axis.

Hyperparameter	Value
Learning rate	6e-4
Num. Environment	1024
Optimizer	Adam
Normalization	Layernorm
Dropout	0

Table 7. Hyper-parameters for Student's Training Algorithm

Hyperparameter	Value
Input Size	(512, 3)
Key points $\mathbf{C_i}$	[64, 16]
Grouping Neighbours $\mathbf{K}$	32
Grouped feature $\mathbf{M_i}$	[32, 128]
Global points MLP	MLP(4096, 1024, 1024, 4096)
History length	5
History Decoder	Conv1d+MaxPool
History Decoder channel	128
FiLM block Num	3
Pose Predictor Shared	MLP(4096, 256)
Translation predictor	MLP(256, 128, 64, 3)
Rotation predictor	MLP(256, 128, 64, 3)
Actor	MLP(4736, 1024, 256)

Table 8. Hyper-parameters for Student's Encoder and Policy

Hyperparameter	Value
RL algorithm	PPO
Adam stepsize	3e-4
Num. Environment	4096
GAE parameter	0.95
Discount Factor	0.99
PPO clip range	0.3
Num. epoch	8

Table 9. Hyper-parameters for Teacher's PPO Algorithm

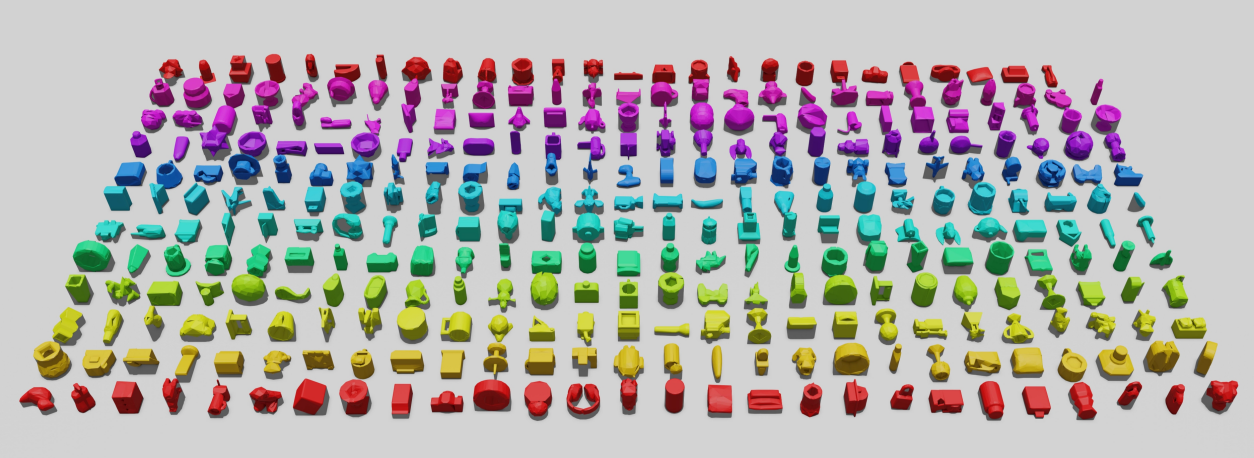

Figure 10. Objects used for training in the simulation benchmark.

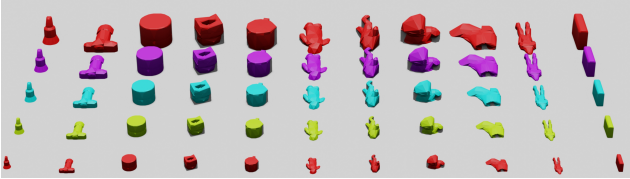

Figure 11. Unseen objects used for evaluation in the simulation benchmark.

Hyperparameter	Value
Key points $\mathbf{C_i}$	[16]
Grouping Neighbors $\mathbf{K}$	32
Grouped feature Channels $\mathbf{M_i}$	[128]
Shared MLP	MLP(512, 256, 128)
Actor	MLP(64, 20)
Critic	MLP(64, 1)

Table 10. Hyper-parameters for Teacher's Encoder and Policy

10. Vision Encoder

We use a simplified version of PointNet++ [41] as our point cloud encoder. Specifically, the student policy encoder employs two layers of fixed-scale grouping, while the teacher policy encoder uses one layer. In the $\mathbf{i^{th}}$ grouping layer, $\mathbf{C_i}$ key points are selected via farthest point sampling (FPS), and each key point forms a group with its $\mathbf{K}$ nearest neighbors (KNN). Each cluster is then processed by two per-point MLP layers and two global MLP layers to generate a group feature. The output point cloud consists only of the $\mathbf{C_i}$ selected key points, each enriched with its corresponding $\mathbf{M_i}$-dimensional group feature. After the grouping layers, the per-point features are concatenated and passed through residual MLP layers, following the structure of PointNet++. The final output consists of several point tokens with grouped features.

11. Hyper-parameters

The following Tables 7, 8, 9, 10 demonstrate the hyper-parameters of our policy and the teacher policy.

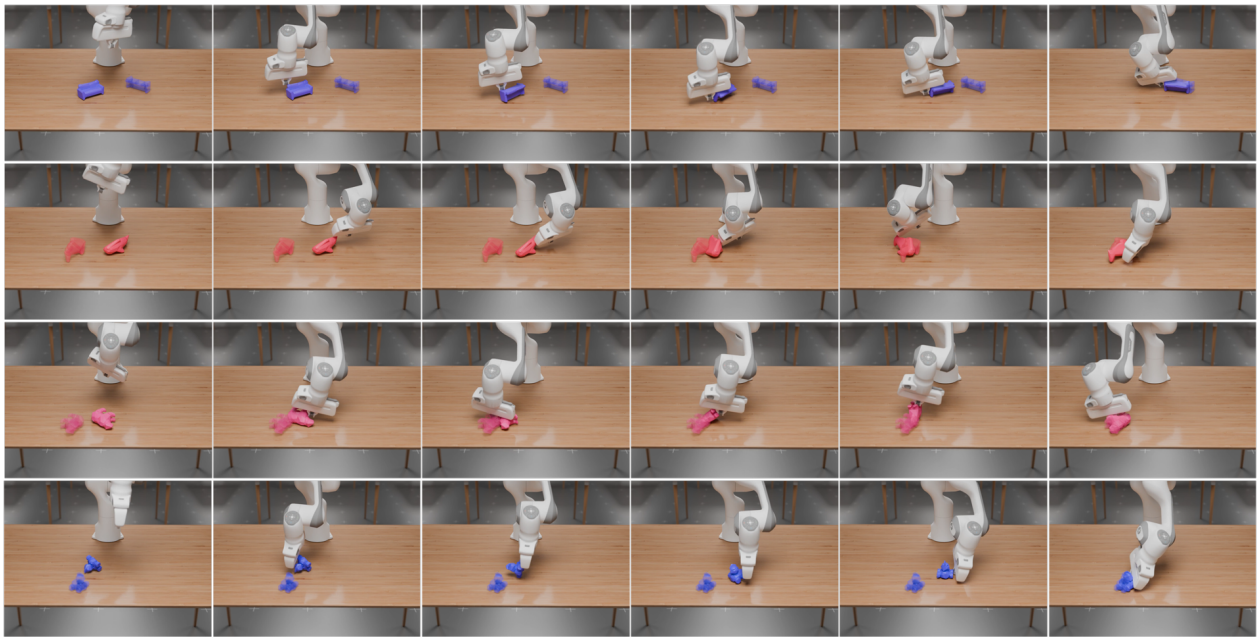

Figure 12. Qualatative results in the simulation.