

A Data Augmentation Loop

A.1 Pseudo Code

Algorithm 1 DATA AUGMENTATION LOOP

Require: Human data $D = (G, a^W)$, training set $D_t \in D$, high-level planner π^H , low-level controller π^L , augmentation iteration L , data augment function $L_{aug}()$, wrist pose trajectories $G_t = (g_t, g_{t+1}, \dots, g_{t+T-1})$, goal trajectory of the object and its geometric features $a_t^W = (a_t^W, a_{t+1}^W, \dots, a_{t+T-1}^W)$.

- 1: Initialize $\pi^H, \pi^L, D_t = \{\}$.
- 2: **for** iteration $m = 0, 1, \dots, L$ **do**
- 3: **while** until convergence of π^H **do**
- 4: Generate augmented data $L_{aug}(D)$
- 5: Append into training set $D_t \leftarrow D_t + L_{aug}(D)$
- 6: Train π^H on D_t
- 7: **end while**
- 8: **while** until convergence of π^L **do**
- 9: Train π^L on $(\pi^H(G), G)$
- 10: **end while**
- 11: Rollout success trajectories $D_s^t = (a_t^W, G_t)$ with π^L
- 12: Append into human data $D \leftarrow D + D_s^t$
- 13: **end for**

A.2 Detail of the Data Augmentation Loop

Below are the details for each augmentation. The unit of length is centimeters and the unit of angle is degrees.

- Random the object’s mesh scales with a small scale:
 - The scale of the width of the manipulated object ranges from 0.9 to 1.1.
 - The scale of the length of the manipulated object ranges from 0.9 to 1.1.
 - The scale of the height of the manipulated object ranges from 0.9 to 1.1.
- Random the object’s initial pose with a small scale:
 - The x-coordinate of the manipulated object ranges from -0.02 to 0.02.
 - The y-coordinate of the manipulated object ranges from -0.02 to 0.02.
 - The manipulated object’s z-axis Euler degree ranges from 0 to 30.
- Modify the goal trajectories of the object with waypoint interpolation:
 - The x-coordinate of the goal trajectories position is added by ranges from -0.02 to 0.02.
 - The y-coordinate of the goal trajectories position is added by ranges from -0.02 to 0.02.
 - The z-coordinate of the goal trajectories position is added by ranges from -0.02 to 0.02.

B Detail Implementation of RL in Simulation

B.1 Observation Space

Table.5 gives the specific information of the observation space.

B.2 Reward Design

Denote the \hat{g}_i^R , \hat{g}_i^T and \hat{g}_i^J is the current 3D translation, 3D rotation and joint angle of the object respectively, the desired object 3D rotation g_i^R , the desired object 3D translation g_i^T , and the desired

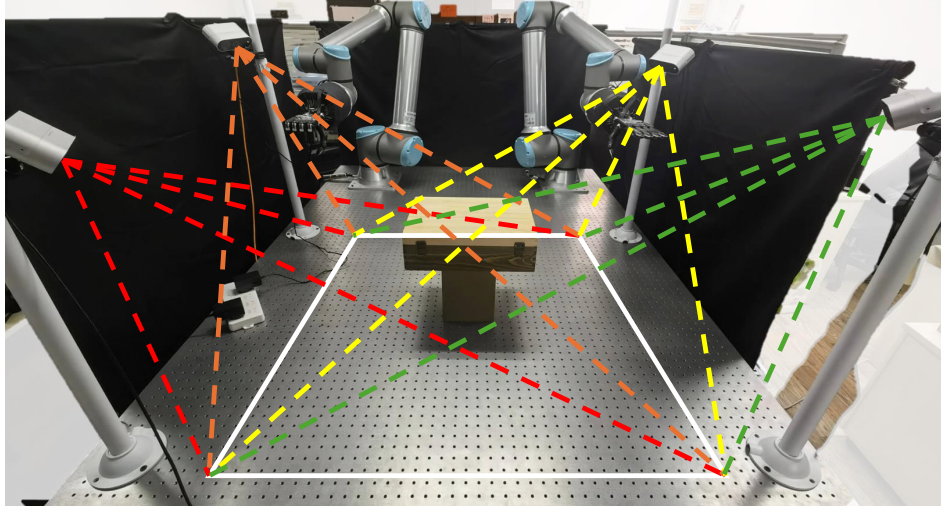


Figure 6: Setup of the cameras.

Index	Description
0 - 60	right arm-hand dof position, velocity
60 - 120	left arm-hand dof position, velocity
120 - 133	right hand end-effector position, velocity, linear velocity, angle velocity
133 - 146	left hand end-effector position, velocity, linear velocity, angle velocity
146 - 159	object base position, rotation, linear velocity, angle velocity
159 - 172	articulated object top part position, rotation, linear velocity, angle velocity
172 - 185	articulated object bottom part position, rotation, linear velocity, angle velocity
185 - 187	object dof position, velocity
187 - 257	desired object motion trajectory $G = (g_t, g_{t+1}, \dots, g_{t+T})$
257 - 397	sequence of 6-DoF wrist actions $(a_t^W, a_{t+1}^W, \dots, a_{t+T}^W)$ generated by high-level planner
397 - 462	right hand fingertip pose, linear velocity, angle velocity
462 - 527	left hand fingertip pose, linear velocity, angle velocity

Table 5: Observation space of our framework in simulation.

object joint angle g_i^J . λ_1 , λ_2 and λ_3 is the hyperparameters to balance the weight of each component of the reward.

The reward function is defined as:

$$r_t = \exp^{-(\lambda_1 * \|g_t^R - \hat{g}_t^R\|_2 + \lambda_2 * \|g_t^T - \hat{g}_t^T\|_2 + \lambda_3 * \|g_t^J - \hat{g}_t^J\|_2)} \quad (1)$$

where $\lambda_1 = 20$, $\lambda_2 = 1$, and $\lambda_3 = 5$.

We use an exponential map in the reward function, which is an effective reward shaping technique used in the case to minimize the distance, introduced by [68, 69]. To improve the calculation efficiency, we use quaternion to represent the object orientation. The angular position difference is then computed through the dot product between the normalized goal quaternion and the current object’s quaternion.

C Detail Implementation in Real-World

C.1 Perception

Our perception setup is shown in Figure 6. We arranged 4 identical Femto Bolt cameras around the table and face towards the object. We use FoundationPose [67] to estimate the articulated object pose. To remove the abnormal results, we compare each pose to the desired pose and remove the pose if the

error is smaller than a threshold (5 centimeters in translation and 0.5 radians in orientation). Finally, we average the rest of the poses as our observation for the policy. If none of the poses is smaller than the threshold, we continue to use the pose from the previous frame.

C.2 Policy Distillation

We use the DAgger [70] algorithm for policy distillation. Table.6 gives the specific information of the observation space of the distilled policy.

Index	Description
0 - 24	right hand dof position
24 - 48	left hand dof position
48 - 55	right hand end-effector position, rotation
55 - 62	left hand end-effector position, rotation
62 - 69	articulated object top part position, rotation
69 - 76	articulated object bottom part position, rotation
76 - 77	object dof position
77 - 147	desired object motion trajectory $G = (g_t, g_{t+1}, \dots, g_{t+T})$
147 - 287	sequence of 6-DoF wrist actions $(a_t^W, a_{t+1}^W, \dots, a_{t+T}^W)$ generated by high-level planner

Table 6: Observation space of our framework in the real-world.

D Hyperparameters of the PPO

Table.7 gives the hyperparameters of the PPO.

Hyperparameters	Value
Num mini-batches	4
Num opt-epochs	5
Num episode-length	8
Hidden size	[1024, 1024, 512, 256]
Clip range	0.2
Max grad norm	1
Learning rate	3.e-4
Discount (γ)	0.998
GAE lambda (λ)	0.95
Init noise std	0.8
Desired kl	0.02
Ent-coef	0

Table 7: Hyperparameters of PPO.

E Domain Randomization

Isaac Gym provides lots of domain randomization functions for RL training. We add the randomization for all the tasks as shown in Table. 8 for each environment. we generate new randomization every 1000 simulation steps.

Parameter	Type	Distribution	Initial Range
Robot			
Mass	Scaling	uniform	[0.5, 1.5]
Friction	Scaling	uniform	[0.7, 1.3]
Joint Lower Limit	Scaling	loguniform	[0.0, 0.01]
Joint Upper Limit	Scaling	loguniform	[0.0, 0.01]
Joint Stiffness	Scaling	loguniform	[0.0, 0.01]
Joint Damping	Scaling	loguniform	[0.0, 0.01]
Object			
Mass	Scaling	uniform	[0.5, 1.5]
Friction	Scaling	uniform	[0.5, 1.5]
Scale	Scaling	uniform	[0.95, 1.05]
Position Noise	Additive	gaussian	[0.0, 0.02]
Rotation Noise	Additive	gaussian	[0.0, 0.2]
Observation			
Obs Correlated. Noise	Additive	gaussian	[0.0, 0.001]
Obs Uncorrelated. Noise	Additive	gaussian	[0.0, 0.002]
Action			
Action Correlated Noise	Additive	gaussian	[0.0, 0.015]
Action Uncorrelated Noise	Additive	gaussian	[0.0, 0.05]
Environment			
Gravity	Additive	normal	[0, 0.4]

Table 8: Domain randomization of all the tasks.