

A Full Environment and Evaluation Protocol Details

In this section, we provide detailed descriptions of our SIMPLER environments along with our simulation and real-world evaluation protocols.

For the RT-1 Robot, we adopt the following language-conditioned tasks:

- **“pick coke can”**. The robot is instructed to grasp the empty coke can on the table and lift it up. In the default setting, no distractors are added to the scene. We place the coke can in 3 different orientations: horizontally laying, vertically laying, and standing. For each orientation, we place the coke can at 25 grid positions within a rectangle on the tabletop, yielding 25 trials per orientation and 75 trials in total.
- **“move {obj1} near {obj2}”**. We place a triplet of objects on the tabletop in a triangle pattern. In each trial, one object serves as the source object, one serves as the target, and the other serves as the distractor (this creates 6 trials for each triplet and each triangle pattern). We randomly choose 5 triplets of objects among a total of 8 objects (blue plastic bottle, pepsi can, orange, 7up can, apple, sponge, coke can, redbull can), and adopt 2 triangle patterns (upright and inverted). This creates a total of $5 \times 2 \times 6 = 60$ trials. The 5 triplets chosen are:
 - blue plastic bottle, pepsi can, orange
 - 7up can, apple, sponge
 - coke can, redbull can, apple
 - sponge, blue plastic bottle, 7up can
 - orange, pepsi can, redbull can
- **“(open / close) (top / middle / bottom) drawer”**. The robot is positioned in front of a cabinet that contains 3 drawers and instructed to open / close a specific drawer, testing its ability to manipulate articulated objects. We place the robot at 9 grid positions within a rectangle on the floor, yielding a total of $9 \times 3 \times 2 = 54$ trials.
- **“open top drawer; place apple into top drawer”**. The robot opens the top drawer and places the apple from the cabinet top into the top drawer, testing its ability to perform longer-horizon tasks. We place the robot at 3 different positions on the floor and the apple at 9 different positions within a grid on the cabinet top, yielding a total of $3 \times 9 = 27$ trials. Initially, the policies receive the “open top drawer” instruction. We switch to the “place apple into top drawer” instruction once the robot outputs the “terminate” token or after half of the time limit has elapsed.

For the WidowX + Bridge (with WidowX-250 6DOF robot), we adopt the following tasks:

- **“put the spoon on the towel”**. We place the spoon on a vertex of a square (with edge length 15cm) on the tabletop, and we place the towel on another vertex. The spoon’s initial orientation switches between horizontal and vertical, requiring the robot to perform gripper reorientation. This creates a total of $2 \times 12 = 24$ trials.
- **“put carrot on plate”**. We adopt a similar setup as “put the spoon on the towel”, replacing the spoon with carrot and the towel with plate.
- **“stack the green block on the yellow block”**. We place a green block on a vertex of a square on the tabletop, and we position a yellow block on another vertex. The block dimensions are 3cm. We also adopt two differently-sized squares (edge length 10cm and 20cm). This creates a total of $2 \times 12 = 24$ trials.
- **“put eggplant into yellow basket”**. We place an eggplant on the right basin of a sink, and we place a yellow basket on the left basin. The eggplant is dropped into the sink at a random position and orientation, and we ensure that the eggplant is directly graspable (i.e., not too close to the edges of the sink basin). We perform a total of 24 trials.

Algorithm 1 RT-1 Robot Controller in Simulation

Require: (1) Current end-effector action (\mathbf{x}_a, R_a) , along with sensed arm joint positions and velocities $q_{\text{arm}}, v_{\text{arm}}$; (2) Current gripper action g_a , along with sensed gripper joint position and velocity $q_{\text{grip}}, v_{\text{grip}}$; (3) Simulation frequency H_{sim} (501 in our implementation), action output frequency (control frequency) H_{ctrl} (3 in our implementation following [1]); (4) Arm velocity, acceleration, and jerk limits L_{arm} (equal to 1.5, 2.0, 50.0 respectively); (5) Gripper velocity, acceleration, and jerk limits L_{grip} (equal to 1.0, 7.0, 50.0 respectively); (6) Current action timestep T within an episode; (7) A planner that takes goal and initial joint positions and velocities as input (along with velocity, acceleration, and jerk constraints), and outputs a time-parametrized trajectory.

```
1: # Arm motion planning
2:  $(\mathbf{x}, R) = \text{ForwardKinematics}(q_{\text{arm}})$ 
3:  $(\mathbf{x}_{\text{goal}}, R_{\text{goal}}) = (\mathbf{x}_a + \mathbf{x}, R_a \cdot R_{\text{arm}})$ 
4:  $(q_{\text{goal}}, v_{\text{goal}}) = (\text{InverseKinematics}(\mathbf{x}_{\text{goal}}, R_{\text{goal}}, q_{\text{arm}}), 0.0)$ 
5:  $\text{ArmPlan} = \text{Planner}(q_{\text{goal}}, v_{\text{goal}}, q_{\text{arm}}, v_{\text{arm}}, L_{\text{arm}})$ 
6: # Gripper motion planning
7: if  $T = 0$  then ▷ At the beginning of episode
8:    $q_{\text{lastplan}, \text{grip}}, v_{\text{lastplan}, \text{grip}} = q_{\text{grip}}, 0.0$ 
9:    $q_{\text{lastgoal}, \text{grip}} = q_{\text{grip}}$ 
10: end if
11: if  $|g_a| < 0.01$  then ▷ Small action filtering
12:    $q_{\text{goal}, \text{grip}} = q_{\text{lastgoal}, \text{grip}}$ 
13: else
14:    $q_{\text{goal}, \text{grip}} = q_{\text{lastplan}, \text{grip}} + g_a$ 
15: end if
16:  $v_{\text{goal}, \text{grip}} = 0.0$ 
17:  $\text{GripPlan} = \text{Planner}(q_{\text{goal}, \text{grip}}, v_{\text{goal}, \text{grip}},$ 
    $q_{\text{lastplan}, \text{grip}}, v_{\text{lastplan}, \text{grip}}, L_{\text{grip}})$ 
18: # Execute arm and gripper plans at each simulation step
19: for each  $i = 1 \dots \frac{H_{\text{sim}}}{H_{\text{ctrl}}}$  do
20:    $t = \frac{i}{H_{\text{sim}}}$ 
21:    $q_{\text{lastplan}, -} = \text{ArmPlan}(t)$ 
22:    $\text{SetArmJointPosTarget}(q_{\text{lastplan}})$ 
23:    $q_{\text{lastplan}, \text{grip}}, v_{\text{lastplan}, \text{grip}} = \text{GripPlan}(t)$ 
24:    $\text{SetGripperJointPosTarget}(q_{\text{lastplan}, \text{grip}})$ 
25:    $\text{SetGripperJointVelTarget}(v_{\text{lastplan}, \text{grip}})$ 
26: end for each
27:  $q_{\text{lastgoal}, \text{grip}} = q_{\text{goal}, \text{grip}}$ 
28:  $T = T + 1$ 
```

631 For Octo simulated evaluations, since the model involves a non-deterministic diffusion head, we
632 average its success rates across three different random seeds to produce a lower-variance estimate of
633 the policy’s simulation performance. Additionally, for RT-1 Robot simulated evaluations, we aver-
634 age results over four versions of robot arm and gripper colors to account for changes in arm texture
635 during real robot rollouts (see Section 4.2). For the WidowX environments, given the consistent
636 black color of the arm and gripper across videos, we skip this step.

637 The number of evaluation trials we present above pertain to the real-world evaluation setup. For
638 our “Variant Aggregation” simulation evaluation setup, the number of trials is multiplied by the
639 number of simulation environment variants. For our “Visual Matching” simulation evaluation setup,
640 the number of trials is multiplied by the number of tuned robot arm colors for the RT-1 Robot
641 evaluation setup, along with the number of seeds for the Octo policies.

642 B More Implementation Details of Our Real-to-Sim Evaluation System

643 B.1 Robot Controllers

644 **RT-1 Robot** Given translation, rotation, and gripper action output from a model, we adopt Algo-
645 rithm 1 in simulator to execute the action commands. The simulation frequency in the algorithm
646 refers to the number of simulation steps per second, while the control frequency refers to the num-

Algorithm 2 WidowX Controller in Simulation

Require: (1) Current end-effector action (\mathbf{x}_a, R_a) , along with sensed arm joint positions q_{arm} ; (2) Current gripper action g_a , along with sensed gripper joint position q_{grip} ; (3) Simulation frequency H_{sim} (500 in our implementation), action output frequency (control frequency) H_{ctrl} (5 in our implementation following); (4) Current action timestep T within an episode; (5) A function S that maps a \mathbb{R}^3 position vector and a 3×3 $\mathbb{SO}(3)$ rotation matrix to a 4×4 $\mathbb{SE}(3)$ matrix.

```
1: if  $T = 0$  then                                     ▷ At the beginning of episode
2:    $q_{\text{lastgoal}} = q_{\text{arm}}$ 
3: end if
4:  $(\mathbf{x}, R) = \text{ForwardKinematics}(q_{\text{lastgoal}})$ 
5:  $(\mathbf{x}_{\text{goal}}, R_{\text{goal}}) = S^{-1}(S(\mathbf{x}, I) \cdot S(\mathbf{x}_a, R_a) \cdot$   

    $S(-\mathbf{x}, I) \cdot S(\mathbf{x}, R_{\text{arm}}))$ 
6:  $q_{\text{goal}} = \text{InverseKinematics}(\mathbf{x}_{\text{goal}}, R_{\text{goal}}, q_{\text{arm}})$ 
7:  $q_{\text{goal, grip}} = g_a$ 
8:  $\text{SetArmJointPosTarget}(q_{\text{goal}})$ 
9:  $\text{SetGripperJointPosTarget}(q_{\text{goal, grip}})$ 
10:  $q_{\text{lastgoal}} = q_{\text{goal}}$ 
11:  $T = T + 1$ 
```

ber of control commands (policy action outputs) per second. We use the open-source library Ruckig¹ for time-optimal joint motion planning with velocity, acceleration, and jerk constraints. Note that the duration of planned trajectories may exceed the interval between two control commands.

WidowX We present our WidowX controller implementation in Algorithm 2.

B.2 Robot and Object Assets

Robots For RT-1 Robot, we convert the publically-released MuJoCo `.mjcf` robot description to URDF robot description. We also refine the collision mesh of the robot base link from the original assets to prevent erroneous mesh penetrations. For WidowX, we directly export the URDF robot descriptions from the official Interbotix repository using ROS. To simulate the RT-1 Robot, we find that the Projected Gauss-Seidel solver in PhysX causes mesh penetration behaviors during the process of object grasping. Thus, we enable the Temporal Gauss-Seidel solver in both SAPIEN and Isaac Sim’s simulation backends to produce correct grasping behaviors.

The RT-1 Robot uses a customized egocentric camera mounted on the robot head, while the WidowX + Bridge V2 setup uses a Logitech C920 third-view camera. We use known robot camera intrinsics if possible, and when they are unknown, we obtain them from real evaluation video frames using efficient interactive GUI tools such as fSpy.

Objects We adopt the following procedure to obtain object assets. Except creating precise models for articulated objects like cabinets, the process does not involve heavy manual effort.

- Obtain raw 3D object models from public repositories (e.g., Objaverse [60]), from 3D scanning of objects purchased from Amazon, from single-view 3D generation (e.g., One-2-3-45++ [61]), or from manual modeling based on precise measurements of real-world counterparts (we only used the last technique for articulated objects like cabinets since this requires the most human effort; we highlight the acceleration of articulated asset curation process through approaches like multi-view [62] or interactive [63] articulated object generation as an avenue for future work).
- Process 3D object models in Blender such that the dimensions of objects are similar to those used in the real world, and that the object meshes do not contain too many vertices (to limit the sizes of object meshes).
- Optionally, use our Visual Matching approach (see below) to improve the texture of 3D object models.

¹<https://github.com/pantor/ruckig>

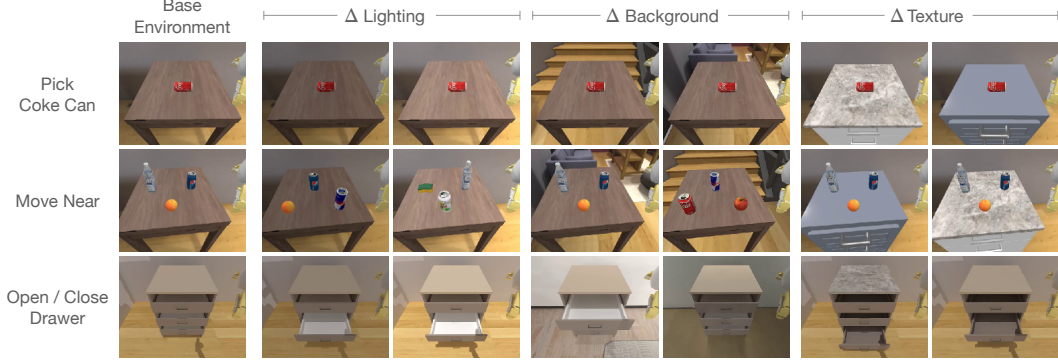


Figure 5: Subset of environment variations under our “Variant Aggregation” evaluation setup, visualized in SAPIEN from RT-1 Robot’s egocentric view. The variations cover different lightings, backgrounds and table textures and are modified from ReplicaCAD [17] scenes.

- Export visual mesh and collision mesh of objects. For collision mesh, further perform CoACD [64] to obtain watertight and locally convex collision meshes. Optionally, simplify the resulting collision mesh and perform minor modifications using Blender (e.g., make the bottom of cans or bottles flat).
- Set the object to have a simple uniform density by querying their common material density in GPT-4 or google search, or (for objects with non-uniform densities like empty coke can), querying their mass and dividing by their visual mesh volume.

To perform visual matching of object textures, we adopt the following steps: (1) Crop the target object in a real image using SAM [56]; (2) Perform a *coarse* estimation of object pose by importing it into the simulation and adjusting its position such that its simulation segment mask overlaps with the real one; (3) Employ differential rendering (using Nvdiffrast) to optimize the simulation asset’s pose such that it *precisely aligns* with the real object’s segmentation mask; (4) “Unproject” the real object’s RGB texture values onto the simulation object mesh; (5) Optionally, generate the remaining views of the object through a diffusion model (Zero123++ [65]), and refine the poses of novel views using a rendering loss with the existing object view. Finally, unproject the novel view textures onto the simulation object mesh. This whole process is semi-automatic, and can thus be completed efficiently. We commit to release a convenient command-line python script for this process.

B.3 SIMPLER-Variant Aggregation

A common approach for addressing visual gaps in sim-to-real policy training is domain randomization. By performing training across a range of randomized parameters, such as textures and lighting, prior works aim to obtain policies that are robust to visual distribution shifts in the real-world [40, 41]. Similarly, in real-to-sim evaluation, we can aggregate evaluation results across a range of visual simulator characteristics to obtain a more faithful signal for the policy’s performance. In practice, we implement this SIMPLER-“Variant Aggregation” approach as an alternative to SIMPLER-Visual Matching, described in Section 4.2. Concretely, we create a “base” version of our simulation environment and then creating “variants” of this environment along four axes of visual variation: background, lighting, distractors, and table texture. For each axis, we construct 2 variations of the base setup similar to [59], covering backgrounds from different rooms, lighter and darker lighting, fewer and more distractors, and solid color and complex table textures. We visualize an example of such simulator variations for various table-top tasks in Fig. 5. We average policy performance in simulation across all variants of an environment to obtain our final performance estimate.

RT-1 Robot Evaluation Setup	Policy	Pick Coke Can				Move Near	Open / Close Drawer			Open Top Drawer and Place Apple
		Horizontal Laying	Vertical Laying	Standing	Average	Average	Open	Close	Average	Average
Real Eval	RT-1 (Converged)	0.960	0.880	0.720	0.853	0.633	0.815	0.926	0.870	0.185
	RT-1 (15%)	1.000	0.960	0.800	0.920	0.583	0.704	0.889	0.796	0.185
	RT-1-X	0.880	0.560	0.840	0.760	0.450	0.519	0.741	0.630	0.407
	RT-2-X	0.920	0.800	1.000	0.907	0.733	0.333	0.630	0.481	0.074
	Octo-Base	0.440	0.200	0.240	0.293	0.350	0.148	0.519	0.333	0.000
	RT-1 (Begin)	0.200	0.000	0.200	0.133	0.017	0.000	0.000	0.000 ²	0.000
SIMPLER Eval (Variant Aggregation)	RT-1 (Converged)	0.969	0.760	0.964	0.898	0.500	0.270	0.376	0.323	0.026
	RT-1 (15%)	0.920	0.704	0.813	0.813	0.446	0.212	0.323	0.267	0.021
	RT-1-X	0.569	0.204	0.698	0.490	0.323	0.069	0.519	0.294	0.101
	RT-2-X	0.822	0.754	0.893	0.823	0.792	0.333	0.372	0.353	0.206
	Octo-Base	0.005	0.000	0.013	0.006	0.031	0.000	0.021	0.011	0.000
	RT-1 (Begin)	0.022	0.013	0.031	0.022	0.040	0.005	0.132	0.069	0.000
	MMRV↓	0.093	0.133	0.140	0.084	0.111	0.303	0.321 ³	0.321	0.148
	Pearson r ↑	0.947	0.937	0.933	0.960	0.887	0.629	0.613	0.737	0.235
SIMPLER Eval (Visual Matching)	RT-1 (Converged)	0.960	0.900	0.710	0.857	0.442	0.601	0.861	0.730	0.065
	RT-1 (15%)	0.860	0.790	0.480	0.710	0.354	0.463	0.667	0.565	0.130
	RT-1-X	0.820	0.330	0.550	0.567	0.317	0.296	0.891	0.597	0.213
	RT-2-X	0.740	0.740	0.880	0.787	0.779	0.157	0.343	0.250	0.037
	Octo-Base	0.210	0.210	0.090	0.170	0.042	0.009	0.444	0.227	0.000
	RT-1 (Begin)	0.050	0.000	0.030	0.027	0.050	0.000	0.278	0.139	0.000
	MMRV↓	0.027	0.027	0.053	0.031	0.111	0.000	0.123	0.055	0.000
	Pearson r ↑	0.981	0.964	0.942	0.976	0.855	0.983	0.768	0.915	0.969

Table 2: Real-world and SAPIEN evaluation results across different policies on RT-1 Robot tasks. We present success rates for the “Variant Aggregation” and “Visual Matching” approaches in Sec. 4.2. We calculate the Mean Maximum Rank Violation (“MMRV”, lower is better) and the Pearson correlation coefficient (“Pearson r ”, higher is better) to assess the alignment between simulation and real-world relative performances across different policies.

WidowX+Bridge Evaluation Setup	Policy	Put Spoon on Towel		Put Carrot on Plate		Stack Green Block on Yellow Block		Put Eggplant in Yellow Basket	
		Grasp Spoon	Success	Grasp Carrot	Success	Grasp Green Block	Success	Grasp Eggplant	Success
Real Eval	RT-1-X	0.042	0.000	0.167	0.000	0.000	0.000	0.033	0.000
	Octo-Base	0.500	0.333	0.500	0.250	0.292	0.000	0.400	0.233
	Octo-Small	0.542	0.417	0.208	0.083	0.583	0.125	0.700	0.433
SIMPLER Eval (Visual Matching)	RT-1-X	0.167	0.000	0.208	0.042	0.083	0.000	0.000	0.000
	Octo-Base	0.347	0.125	0.528	0.083	0.319	0.000	0.667	0.431
	Octo-Small	0.778	0.472	0.278	0.097	0.403	0.042	0.875	0.569
	MMRV↓	0.000	0.000	0.000	0.111	0.000	0.000	0.000	0.000
	Pearson r ↑	0.778	0.827	0.995	0.575	0.964	1.000	0.995	0.990

Table 3: Real-world and SAPIEN simulation evaluation results for the WidowX + Bridge setup. We report both the final success rate (“Success”) along with partial success (e.g., “Grasp Spoon”).

C Full Results for Real-and-Sim Relative Policy Performance Correlation Experiments

In Table 2 and Table 3, we present full evaluation results for our experiments in Sec. 5.2, which demonstrate that SIMPLER environments show strong performance relationship correlations with real-world evaluations.

D Full Results for Real-and-Sim Policy Behavior Correlation Experiments under Environment Distribution Shifts

In Table 4, Table 5, and Table 6, we present full evaluation results for our experiments in Sec. 5.3, which demonstrate that SIMPLER environments show strong policy behavior correlations with real-world evaluations under different environment distribution shifts.

²After running 2 real evaluation trials, robot operators decided that since this policy would potentially damage the robot on the Drawer tasks, the real evaluation was terminated.

³As real evaluation was terminated due to risk of damaging the robot, we expect the MMRV to be less than this number if real evaluation were to continue.

Policy	Distribution Shift	Pick Coke Can			Move Near			Avg.			Real TableTop [59]	
		$ \Delta \text{ Success} $	MMRV \downarrow	$r \uparrow$	$ \Delta \text{ Success} $	MMRV \downarrow	$r \uparrow$	$ \Delta \text{ Success} $	MMRV \downarrow	$r \uparrow$	$ \Delta \text{ Success} $	
RT-1 w/o Aug	Background	0.013			0.083			0.048			0.028	
	Lighting	0.040			0.075			0.057			0.083	
	Distractors	0.027	0.000	0.779	0.133	0.055	0.939	0.080	0.000	0.831	0.111	
	Table Texture	0.113			0.175			0.144			0.389	
	Camera Pose	0.753			0.192			0.473			0.458	
RT-1 +Aug	Background	0.153			0.092			0.123			0.167	
	Lighting	0.033			0.117			0.075			0.042	
	Distractors	0.033	0.041	0.984	0.084	0.125	0.721	0.059	0.041	0.970	0.083	
	Table Texture	0.220			0.159			0.189			0.167	
	Camera Pose	0.613			0.175			0.394			0.375	

Table 4: Impact of various distribution shifts on the tabletop manipulation performance of RT-1 policies trained with and without image augmentation. SIMPLER evaluations accurately track the policies’ robustness to distribution shifts, exhibiting low Mean Maximum Rank Violation (“MMRV”) and high Pearson correlation coefficient (“ r ”) with the real world evaluations [59].

Policy	Robustness Factor	Pick Coke Can	Move Near
RT-1 w/o Aug	Base Setup	0.920	0.467
	Background	0.933/0.907	0.533/0.567
	Lighting	0.960/0.960	0.483/0.600
	Distractors	0.880/0.901	0.600 ^a
	Table Texture	0.867/0.747	0.550/0.200
	Camera Pose	0.053/0.280	0.117/0.433
RT-1 +Aug	Base Setup	0.800	0.383
	Background	0.747/0.547	0.483/0.467
	Lighting	0.760/0.773	0.517/0.483
	Distractors	0.813/0.747	0.467
	Table Texture	0.667/0.493	0.450/0.133
	Camera Pose	0.267/0.107	0.200/0.217

^aThe base setup environment already contains distractors, so we construct environment variants without distractors.

Table 5: Success rates of different out-of-distribution generalization factors on the tabletop manipulation performance of RT-1 policies in the SAPIEN simulator. “a/b” denote results on different environment variants (lighting: darker / brighter; table texture: solid color / contrastively patterned; camera pose: oriented lower / higher).

719 E Full Results for Main Paper Ablation Experiments

720 We present detailed results for our main paper’s ablations in Table 7, Table 8, and Table 9.

Policy	Sim Success Range	Real Success	
		Orig Arm Texture	OOD Arm Texture
RT-1-X	[0.507, 0.653]	0.760	0.520
Octo-Base	[0.000, 0.293]	0.293	0.000

Table 6: Impact of arm textures on the success rates of “Pick Coke Can” task in the SAPIEN simulator (Visual Matching evaluation setup) and in the real-world. The ranges of simulation success rates across multiple (tuned and untuned) robot arm colors can predict policy sensitivity to real-world OOD arm textures.

Components			Open Drawer				Close Drawer			
Background	Drawer	Robot	RT-1 (Converged)	RT-1 (15%)	RT-1-X	MMRV↓	RT-1 (Converged)	RT-1 (15%)	RT-1-X	MMRV↓
Real	Real	Real	0.815	0.704	0.519	N/A	0.926	0.889	0.741	N/A
GreenScreen	Curated	Curated	0.703	0.556	0.333	0.000	0.889	0.667	0.851	0.099
GreenScreen	Curated	Original	0.444	0.444	0.259	0.111	0.741	0.630	0.926	0.173
GreenScreen	Original	Original	0.593	0.519	0.148	0.000	0.852	0.778	0.963	0.173
ReplicaCAD	Curated	Curated	0.407	0.259	0.111	0.000	0.667	0.481	0.778	0.173
ReplicaCAD	Curated	Original	0.630	0.407	0.074	0.000	0.630	0.593	0.667	0.173
ReplicaCAD	Original	Curated	0.556	0.296	0.074	0.000	0.667	0.704	0.815	0.173
ReplicaCAD	Original	Original	0.556	0.333	0.074	0.000	0.704	0.556	0.741	0.173

Table 7: Impact of real-to-sim visual gaps on real-and-sim performance correlations. We report the success rates of 3 different policies on 2 tasks: *Open Drawer* and *Close Drawer*. The settings with the smallest MMRV and the smallest absolute performance gap with the real performance are highlighted. Using a combination of “green-screened” background and curated foreground object and robot assets provides the best correlation.

Coke Can Mass	Gripper Friction Coefficient			
	0.25	0.50	1.0	2.0
10 g	0.957	0.967	0.971	0.978
20 g	0.969	0.975	0.978	0.977
40 g	0.963	0.976	0.976	0.976
80 g	0.962	0.962	0.975	0.990

(a) Pearson r between real and SIMPLER evaluations on the Pick Coke Can task under different settings of can mass and gripper friction coefficient. The MMRV is 0.031 in all cases. The use of empty coke cans follows the setup from the RT-1 Robot demonstration dataset and the RT-1 paper [1].

Cabinet Joint Friction	0.0125	0.025	0.05	0.10	0.15	0.20
MMRV↓	0.055	0.055	0.055	0.055	0.105	0.055
Pearson r ↑	0.930	0.941	0.915	0.923	0.903	0.928

(b) MMRV and Pearson r between real and SIMPLER evaluations on the Open/Close Drawer tasks under different settings of cabinet joint friction.

Table 8: SIMPLER is robust to imprecisely estimated physical simulation parameters such as object mass and friction coefficients, as indicated by the low MMRV and high Pearson r in both ablation studies. We use the 6 policies from our RT-1 Robot experiments in these ablations.

RT-1 Robot Evaluation Setup	Policy	Pick Coke Can			Move Near	
		Horizontal Laying	Vertical Laying	Standing	Avg. Success	Avg. Success
Real Eval	RT-1 (Converged)	0.960	0.880	0.720	0.853	0.633
	RT-1 (15%)	1.000	0.960	0.800	0.920	0.583
	RT-1-X	0.880	0.560	0.840	0.760	0.450
	Octo-Base	0.440	0.200	0.240	0.293	0.350
	RT-1 (Begin)	0.200	0.000	0.200	0.133	0.017
SIMPLER Eval (Isaac, Variant Aggre.)	RT-1 (Converged)	0.418	0.377	0.436	0.410	0.150
	RT-1 (15%)	0.428	0.306	0.590	0.441	0.100
	RT-1-X	0.340	0.182	0.618	0.380	0.125
	Octo-Base	0.015	0.020	0.010	0.015	0.020
	RT-1 (Begin)	0.036	0.040	0.054	0.044	0.000
	MMRV↓	0.096	0.112	0.016	0.064	0.053
	Pearson r ↑	0.961	0.949	0.989	0.973	0.865

Table 9: Real-world and Isaac Sim evaluation results for the RT-1 Robot setup. The findings on Isaac Sim are consistent with the findings on the SAPIEN simulator.

Task	Validation Action MSE	Sim Eval (Visual Matching)
Pick Coke Can	0.412 / 0.464	0.031 / 0.976
Move Near	0.408 / 0.230	0.111 / 0.855
Open / Close Drawer	0.346 / 0.264	0.055 / 0.915
Open Drawer & Place Apple	0.265 / 0.198	0.000 / 0.969
Put Spoon on Towel	0.389 / -0.951	0.000 / 0.827
Put Carrot on Plate	0.194 / -0.342	0.111 / 0.575
Stack Block	0.125 / -0.857	0.000 / 1.000
Put Eggplant in Basket	0.366 / -1.000	0.000 / 0.990

Table 10: MMRV / Pearson correlation comparison between our Visual Matching simulation evaluation approach and the simulation-free approach that assesses the MSE between predicted and ground-truth actions on validation trajectories. For the latter approach, we calculate the MMRV / Pearson correlation between the negative MSE and the real policy performance. Our approach yields significantly better real-and-sim policy performance correlations.

Policy	Avg. Real Success	Avg. Sim Success (Visual Matching)
RT-1 (Converged)	0.853	0.857
RT-1 (15%)	0.920	0.710
RT-1 (Single Task Policy)	0.680	0.403
RT-1-X	0.760	0.567
RT-2-X	0.907	0.787
Octo-Base	0.293	0.170
RT-1 (Begin)	0.133	0.027
MMRV↓		0.027
Pearson r↑		0.959

Table 11: Real-world and simulated evaluation results on the Pick Coke Can task, after adding an RT-1 policy trained solely on the Pick Coke Can demonstrations. Our simulated evaluation remains effective, exhibiting low MMRV and high Pearson correlation coefficient with real evaluations.

F More Experiment Results

F.1 More Ablations

Simulated vs. simulation-free evaluation approaches: To evaluate and select real-world robot manipulation policies, a widely-adopted approach involves calculating the MSE loss between predicted and ground-truth actions on a set of held-out validation demonstration trajectories. We are thus interested in the following question: *Does simulated evaluation produce significantly better real-to-sim relative performance correlation than simulation-free approaches?* We conduct an experiment where we calculate the action-prediction MSE loss on the RT-1 Robot dataset and the Bridge dataset. For the Bridge dataset, we randomly select 25 trajectories from the validation demonstration split. For the RT-1 Robot dataset, as a validation split is not publicly available, we randomly select 25 trajectories from the training demonstrations.

We report the results in Table 10. We find that SIMPLER evaluation produces significantly better correlations between real-and-sim performances across different policies, as highlighted by a substantially-lower MMRV and a substantially-higher Pearson correlation coefficient. Furthermore, as demonstrated in Sec. 5.3 of the main paper, SIMPLER evaluation reveals finegrained policy behavior modes, such as robustness to visual distribution shifts, offering insights beyond policy performance comparisons, unlike simulation-free evaluations.

Is simulated evaluation still effective on single-task policies? Previously in the main paper, we focused our simulated evaluation on policies trained on multi-task datasets, such as the RT-1 Robot RT-1 dataset and the Open-X-Embodiment dataset, which contain $\geq 80k$ demonstrations. In this section, we further ask the question: *Is SIMPLER evaluation still effective on policies trained on smaller-scale data, which are potentially more sensitive to real-to-sim visual and control gaps?* To this end, we conduct an experiment where we train RT-1 only with the “pick coke can” demonstrations and evaluate its real and simulation performance. We also compare the MMRV and the

RT-1 Robot Evaluation Setup	Metric	Pick Coke Can				Move Near		Open / Close Drawer			Open Top Drawer and Place Apple
		Horizontal Laying	Vertical Laying	Standing	Avg. Success	Avg. Success		Open	Close	Avg. Success	Avg. Success
SIMPLER (VisMatch)	Kruskal-#Policy $p < 0.05$	0	0	2	3	3		1	2	2	0

(a)

WidowX+Bridge Evaluation Setup	Metric	Put Spoon on Towel		Put Carrot on Plate		Stack Green Block on Yellow Block		Put Eggplant in Yellow Basket	
		Grasp Spoon	Success	Grasp Carrot	Success	Grasp Green Block	Success	Grasp Eggplant	Success
SIMPLER (VisMatch)	Kruskal-#Policy $p < 0.05$	0	0	0	0	0	0	1	0

(b)

Table 12: For our Visual Matching evaluation approach, we conduct Kruskal-Wallis test to assess whether simulation and real-world policy evaluations exhibit significant distribution shift, even though we do not expect to obtain an exact reproduction of real-world performance.

745 Pearson correlation before and after incorporating this single-task policy into the RT-1 Robot exper-
 746 iments. Results are shown in Table 11. We find that our simulated evaluation effectively reflects the
 747 performance rankings of the newly-added single-task policy, with the MMRV remaining low and
 748 the Pearson Coefficient remaining high. This demonstrates SIMPLER evaluation’s versatility across
 749 policies trained on diverse data scales.

750 F.2 Other Metrics: Kruskal Wallis

751 In our previous analysis, we primarily focused on metrics that measure real-to-sim **relative per-**
 752 **formance** alignment between policies. As we match real-to-sim visual input appearance in our
 753 Visual Matching evaluation approach, it also becomes meaningful to measure the simulation distri-
 754 bution shift of **absolute performance** from real-world evaluations, even though we do not expect
 755 the real-to-sim absolute performances to exactly match. Let the real-world evaluation results of N
 756 policies be $\mathbf{r} = \{\mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_N\}$, where $\mathbf{r}_i = (r_{ij})_{j=1}^{N_{\text{trial}}}$ is the indicator of each trial’s success in
 757 the real-world. Let the corresponding simulation evaluation results be $\mathbf{s} = \{\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_N\}$, where
 758 $\mathbf{s}_i = (s_{ij})_{j=1}^{N_{\text{trial}}}$. We perform *Kruskal-Wallis test* for each individual policy (i.e., between each \mathbf{r}_i and
 759 \mathbf{s}_i) to measure whether simulation evaluations exhibit significant distribution shift from real evalua-
 760 tions. We then report the number of policies with significant distribution shift (which we denote as
 761 “Kruskal-#Policy $p < 0.05$ ”).

762 We present the Kruskal-Wallis results in Tab. 12. We find that with the Visual Matching evalua-
 763 tion approach, the simulation trial success distribution is not significantly different from the real
 764 results ($p \geq 0.05$) across many tasks and policies, demonstrating the effectiveness of our simulation
 765 evaluation tool. We also note that our MMRV and the Kruskal metrics complement each other’s
 766 limitations, with the former providing a real-to-sim relative performance alignment perspective, and
 767 the latter providing an absolute performance alignment perspective.