IV. Experiments

Our proposed approaches form a general recipe that allows for the practical application of RL to solve dexterous manipulation with humanoids. In this section, we show experimental results of task capabilities and ablation studies of each proposed technique. Videos can be found on project website.

A. Real-World and Simulator Setup

We use a Fourier GR1 humanoid robot with two arms and two multi-fingered hands. Each arm has 7 degrees of freedom (DoF). For most experiments, we use the Fourier hands, each of which has six actuated DoFs and five underactuated DoFs. To show cross-embodiment generalization, we include results on the Inspire hands, each with 6 actuated DoFs and six underactuated DoFs. The hardware has substantially different mass properties, surface frictions, finger and palm morphologies, and thumb actuations. Figure 1 shows a visualization of both robot hands. We use the NVIDIA Isaac Gym simulator [36].

Perception. As outlined in Section III-D, we use a combination of dense and sparse object features for policy learning in both simulation and real-world transfer. In the real world, we set up an egocentric-view camera by attaching a RealSense D435 depth camera onto the head of a humanoid robot and a third-view camera by putting another RealSense D435 depth camera on a tripod in front of the robot (illustrated in Figure 1). In simulation, we similarly set up an egocentric-view camera and a third-view camera by calibrating the camera poses against the real camera poses. The *dense* object feature is obtained by directly reading depth observations from the egocentric-view camera. The *sparse* feature is obtained by approximating the object's center-of-mass from the third-view camera, using a similar technique as in Lin et al. [31]. As illustrated in Figure 2, we utilize the Segment Anything Model 2 (SAM2) [46] to generate a segmentation mask for the object of interest at the first frame of each trajectory sequence and leverage the tracking capabilities of SAM2 to track the mask throughout all remaining frames. To approximate the 3D center-of-mass coordinates of the object, we calculate the center position of their masks in the image plane, then obtain noisy depth readings from a depth camera to recover a corresponding 3D position. The perception pipeline runs at 5 Hz to match the neural network policy's control frequency.

B. Task Definition

Grasp-and-reach. In this task, the robot needs to use one hand to grasp a tabletop object, lift it up, and move it to a goal position. At task initialization, a scripted visual module determines which hand is closer to the object and instructs the robot to use that hand. The test objects have varying geometric shapes, masses, volumes, surface frictions, colors, and textures; a visualization of all objects can be found in Figure 1. For each trial, we vary the initial position and orientation of objects, as well as the goal position.

Box lift. In this task, the robot needs to lift a box that is too large to be grasped with a single hand. Box colors, sizes, and masses are varied. For each trial, we randomize the boxes' initial position and orientation about the vertical axis.

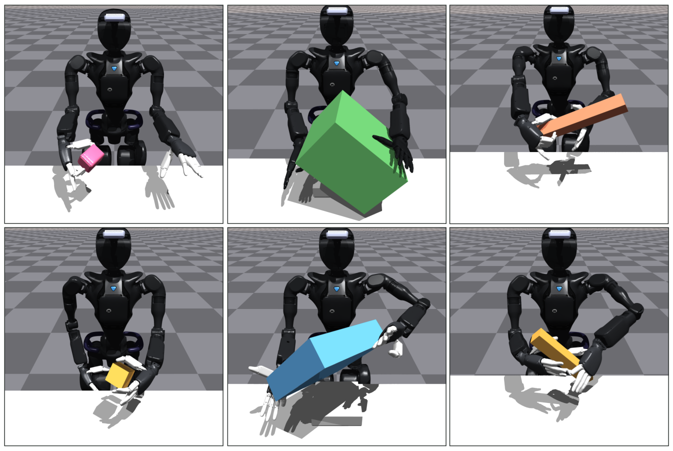
Fig. 3: **Policies learned in simulation.** Left: grasp; middle: box lift; right: bimanual handover (right-to-left, left-to-right).

Autotune MSE	Lowest	Median	Highest
Grasp Success	8 / 10	3 / 10	0 / 10
Reach Success	7 / 10	3 / 10	0 / 10

TABLE I: **Lower MSE from autotune correlates with higher sim-to-real success rate.** For each set of modeling parameters, we test the sim-to-real transfer performance of 10 policy checkpoints (trained identically except for random seed). We evaluate success rate by stages on the `grasp-and-reach` task, and observe a correlation between lower MSE measured by autotune module and higher sim-to-real transfer success rate.

Bimanual handover. In this task, the robot needs to grasp a block object from one side of the table with one hand — that is too far to reach for the other hand — and hand the object over to the other hand. Objects include blocks of varying colors, dimensions, and masses. We vary the initial position and orientation of blocks in each trial.

C. Evaluation of Real-to-Sim Modeling

Effectiveness of autotuned robot modeling. We apply the autotune module outline in Section III-A to find the optimal parameter set for robot modeling. To evaluate its effectiveness, we compare the sim-to-real transfer success rates of three sets of policy checkpoints. All policies are trained with identical task and RL settings, but each set of policies uses robot modeling parameters that achieve different MSE from autotune, ranging from lowest (i.e., smallest real-to-sim gap) to highest (i.e., highest real-to-sim gap). The quantitative results in Table I indicate that autotuned robot modeling improves sim-to-real transfer. Additionally, in our video, we show qualitative results of successful sim-to-real transfer of `grasp-and-reach` policies to the Inspire hands, demonstrating the generalizability of our autotune module.

Effectiveness of approximate object modeling. Empiri-