Details for Track 1

1 Scene Configuration

The real-world robot manipulation environments are shown in Figure 1. It consists of dual leader arms serving as teleoperators, dual follower arms serving as manipulators, and a table with a camera mount. We will run all real-robot tests in this environment.

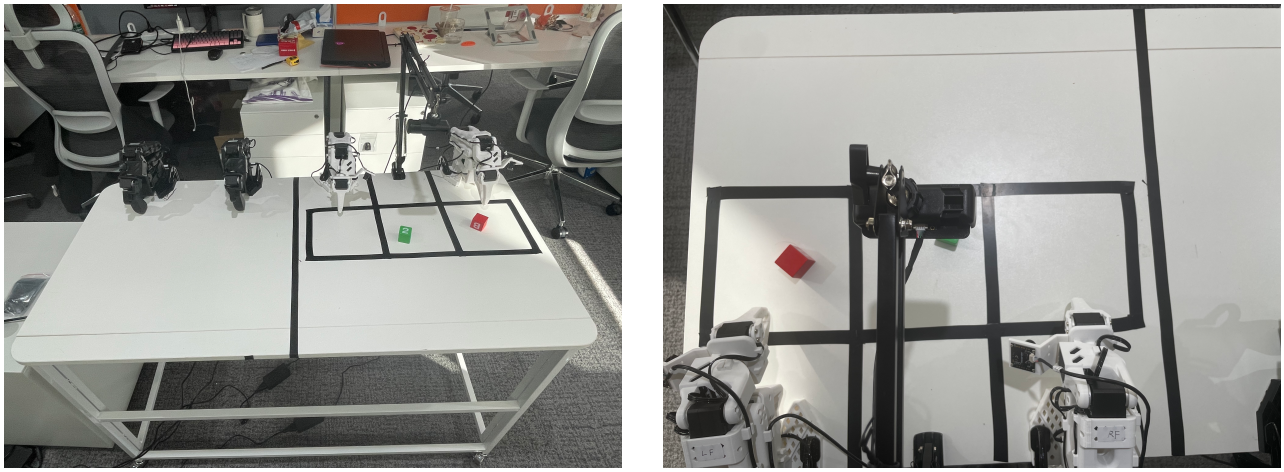

Figure 1: Real-world robot manipulation environment.

Figure 2 shows the position relationship on the tabletop. The numbers are measured manually, so there might be some errors. You can refer to this configuration to build the same scene in the simulation.

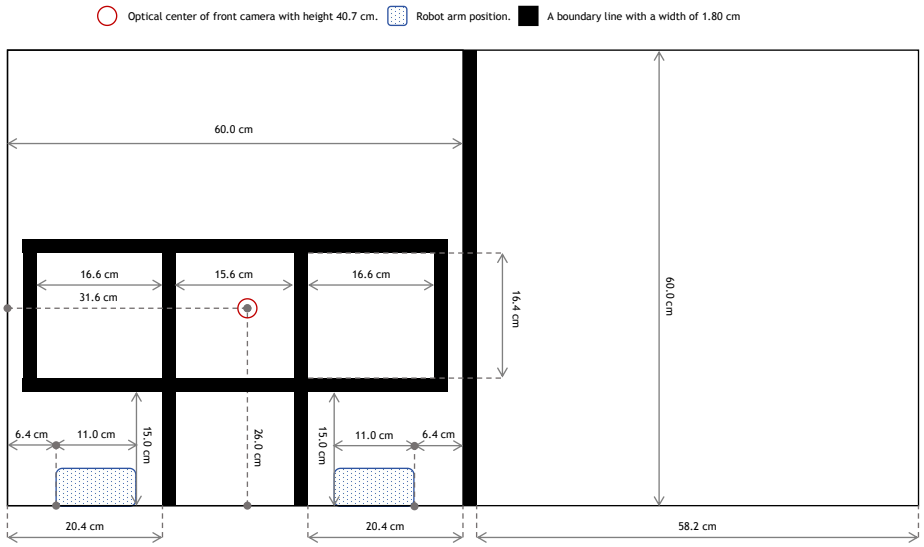

Figure 2: Tabletop Configuration.

Objects in Track 1 are blocks with letters and numbers on both sides. Details are shown in Figure 3. You can create the same blocks in the simulators.

Figure 3: Blocks used in the real world.

2 Tasks Description

Track 1 contains three fundamental robot manipulation tasks designed to assess basic skills in grasping, precise placement, and object differentiation.

2.1 Lift

Lift is a single-arm task for the right arm as shown in Figure 4

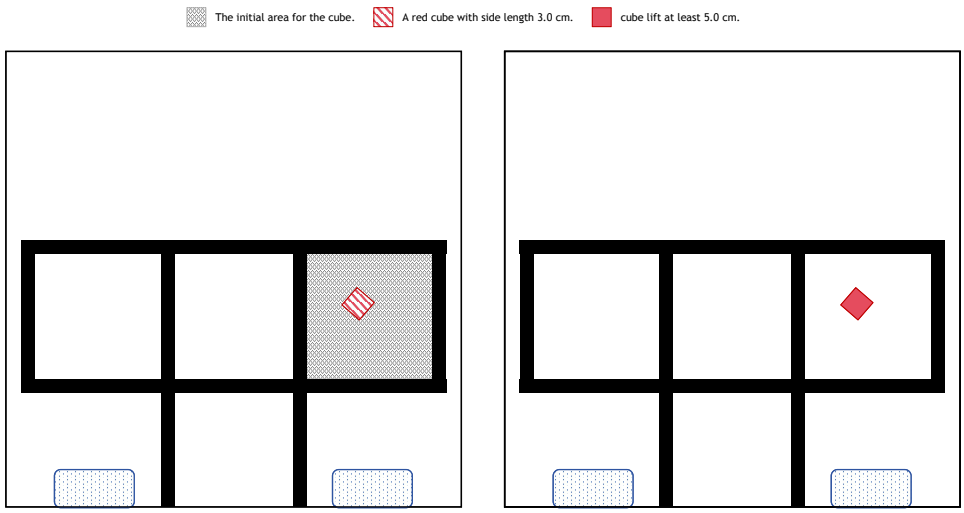

Figure 4: Task *Lift* Configuration. The robot must lift the red cube at least 5.0 cm above the surface.

The task details are shown in following:

- **Objective:** The robot must grasp the red cube and raise it up.

- **Initial State:** A single red cube is placed on the tabletop within a designated starting area.
- **Procedure:** The right arm grasp the red cube and lift up.
- **Success Criteria:** The robot successfully lifts the red cube up to the specified minimum height at least 5 cm.

2.2 Stack

Stack is a single-arm task for the right arm as shown in Figure 5

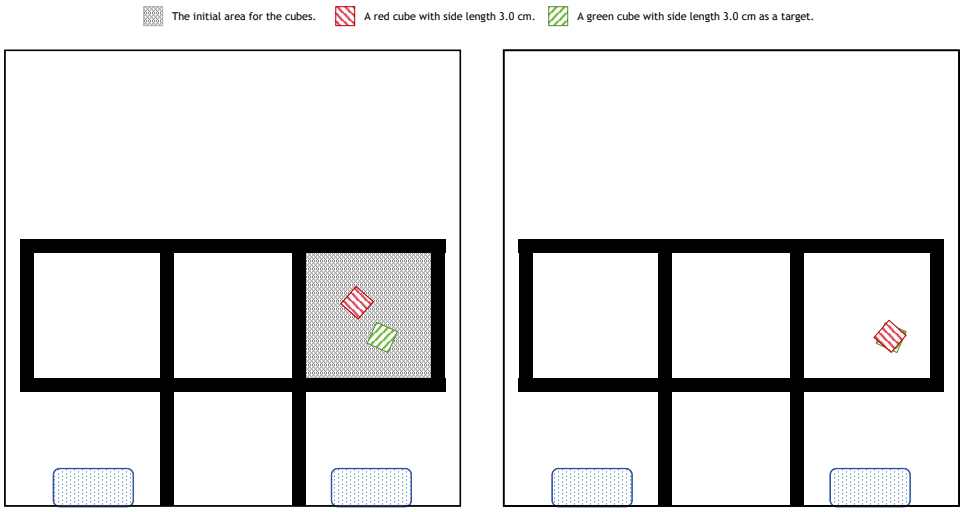

Figure 5: Task *Stack* Configuration. The red cube must be securely stacked on the green cube.

The task details are shown in following:

- **Objective:** The robot must pick up the red cube and place it on top of the green cube.
- **Initial State:** Both the red cube and the green cube are placed within the designated starting area.
- **Procedure:** The right arm grasps the red cube and places it onto the green cube.
- **Success Criteria:** The robot successfully stacks the red cube securely and centrally on top of the green cube, maintaining stability.

2.3 Sort

Sort is a dual-arm task as shown in Figure 6

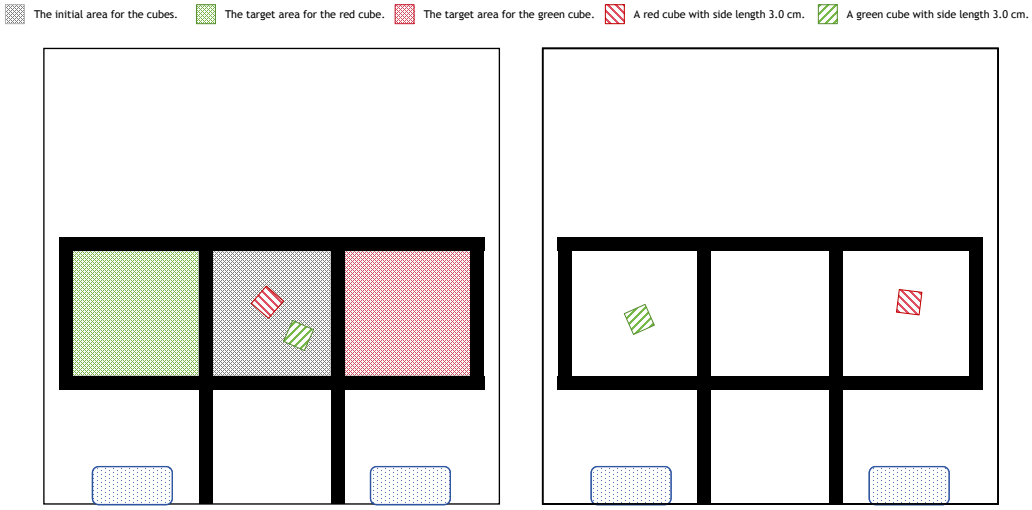

Figure 6: Task *Sort* Configuration. The two cubes must be sorted into their respective target areas.

The task details are shown in following:

- **Objective:** The robot must separate the red and green cubes and move them to their correct target areas.
- **Initial State:** The red cube and the green cube are placed together in a single mixed area.
- **Procedure:** Both arms are used to grasp the cubes and transport them to the target zones.
- **Success Criteria:** The red cube is successfully placed in its designated target area, and the green cube is placed in its designated target area.

3 Real Robot Settings

The real robot is build based on a custom dual-arm setup based on the two SO-ARM100s with sensors.

For real robot setttings, you can refer to `eai-2025-fall-final-project-reference-scripts` on Web Learning.

3.1 URDF Configuration

The script `robot.py` within the reference scripts demonstrates how to correctly load the robot and configure the wrist camera setup within a Sapien simulation environment.

3.2 Control Mode

A critical constraint for the real-robot evaluation is the supported control mode.

- **Real Robot Control:** The real robot only supports **Target Joint Position Control**. Your policy must output a vector of target joint positions for the robot to move to.
- **Simulation Control:** There are no restrictions placed on the control method used in simulation (e.g., velocity control, torque control, end-effector pose control).

3.3 Sensor Configuration

The system is equipped with three vision sensors: two wrist-mounted RGB cameras (one for each arm) and one forward-facing RGB camera.

3.3.1 Wrist Cameras

The two wrist cameras are rigidly attached to the `camera_link` joint of the respective robot arms (refer to `robot.py` for joint attachment details).

- **Resolution:** 480×640 pixels.
- **Vertical Fov:** FOV_y is 50 degrees.

3.3.2 Front Camera

The front camera is manually calibrated. Given its position, it exhibits significant lens distortion.

- **Intrinsic Camera Matrix:** The 3×3 intrinsic matrix is:

$$\begin{pmatrix} 570.21740069 & 0 & 327.45975405 \\ 0 & 570.1797441 & 260.83642155 \\ 0 & 0 & 1 \end{pmatrix}$$

- **Distortion Coefficients:** The five coefficients $[k_1, k_2, p_1, p_2, k_3]$ are:

$$\begin{pmatrix} -0.735413911 \\ 0.949258417 \\ 0.000189059 \\ -0.002003513 \\ -0.864150312 \end{pmatrix}$$

- **Simulation Fidelity:** High distortion is present. While removing distortion (as shown in `undistort.py`) limits the field of view, maintaining fidelity requires adding this distortion to standard simulation camera outputs (refer to `distort.py`). The script `front_camera.py` provides an example of setting up this camera within Sapien.

3.4 Evaluation Interface

For the real-machine evaluation, participants must align their policy with a provided interface, as referenced by the `dummy_eval.py` script.

- **Mandatory Function:** The primary interface is the `get_actions` function, which your policy must implement to output robot control commands based on the current observation.
- **Optional Function:** A `reset` function is also provided for necessary adjustments between evaluation trials.

3.5 Host Machine Configuration

The real-robot evaluation host is a high-performance system. Teams using their own machines for final testing must **contact the TAs in advance**.

- **CPU: AMD Ryzen 9 7945HX** (16 Cores / 32 Threads)
- **RAM: 30 GiB** Total Memory
- **Dedicated GPU: NVIDIA GeForce RTX 4090 Laptop GPU**
- **Dedicated VRAM: 16 GiB**

4 Datasets Description

The demonstration data for Track 1 tasks is standardized using the LeRobot Dataset Format. The dataset includes 50 trajectories for the *Lift* task, and 100 trajectories each for the *Stack* and *Sort* tasks, totaling 250 trajectories.

5 Guidance

For the simulation-only parts of the experiments, participants are only required to ensure consistency with this document regarding the **scene position, initial object range, and success criteria**.

High visual fidelity with the real-world setup is not required for a basic successful submission. If you want to avoid the risks associated with the sim-to-real gap, you may exclusively use the real-world data provided to complete these objectives.

However, if you pursue boosting real-robot success rates through simulation, or aim to achieve sim-to-real bonus points, advanced techniques such as ray tracing, photorealistic rendering, and appropriate data augmentation might be necessary. Should you require further information or assistance with simulation setup, please contact the TAs.