Alex Withers
Braden Eichmeier
Justin Morris

Robot Autonomy Project

Placing Groceries in the Cupboard

Motivation

Many home kitchens are designed to have storage that occupies the full vertical space of the room, a logical decision designed to maximize the efficient use of the available square footage. However, this provides a challenge for people who are disabled, elderly, or otherwise unable to easily access high shelves and cupboards. Robotics provides the potential for a solution, in the form of robotic arms that can access high storage spaces to insert or remove groceries.

Because grocery items vary widely in shape and size, it is practically impossible to manually design behaviors for a robotic arm in such an application that will give it the capacity to handle the full range of items it might encounter. However, a well-designed reinforcement learning scheme could allow the robot to develop robust behaviors for gripping and placing the wide range of items in a grocery-storing application.

One difficulty of reinforcement learning approaches is the requirement to train the system by repeatedly attempting the task. In this particular case, the robot must attempt to grasp groceries and place them in the cupboard hundreds of times as it learns the available behaviors and the associated rewards. Historically, this training has required human intervention to manually reset the task space after each attempt, a process which is tedious and slow. Motivated by the presence of this bottleneck, this project attempts to develop a behavioral loop in which the robot first attempts to execute the RL task of putting groceries into a cupboard, then automatically resets its environment by removing items from the cupboard and returning them to random configurations in the staging area where the items originated.

Overview of Approach

This project utilizes a hybrid systems approach to segment the forward and resetting tasks. A forward pass begins as the robot randomly selects a target grocery item from the table and positions the end effector directly above the item, with the gripper down. Then, the first RL agent attempts to grasp the object. If the grasp is successful, the robot manually stages the item in front of the cupboard for placement. Finally, the second RL agent attempts to place the item inside of the cupboard.

The resetting task operates as a third agent in the hybrid system. Once either of the learning tasks fails to execute, the learning agent analyzes the environment to determine if any items are in the cupboard. The robot then individually removes each item and stages it on the table for the next training episode. Figure 1 shows a graphical representation of the system.

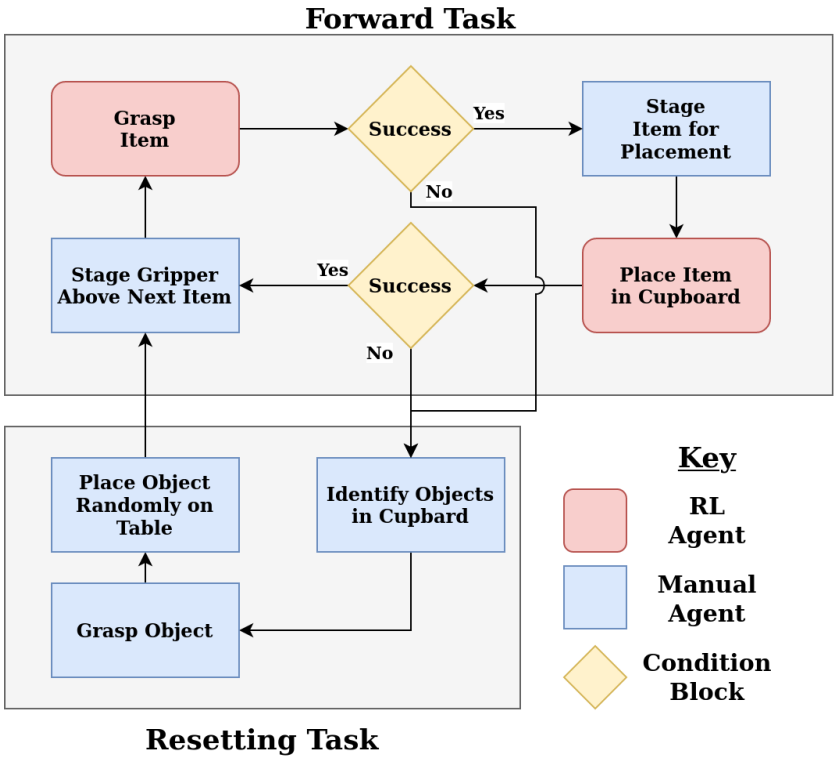

Fig. 1: Manual and learning agent interactions for the groceries task.

Key Challenges

Narrow Behavior Under the Cupboard

All the spawning locations for the cupboard in the simulation environment were directly above the area on the tabletop where the groceries could spawn. Consequently, the robot arm needed to generate a path to travel underneath the cupboard in order to pick up some of the groceries. When aligned with the world Z axis, the robot's end effector took up more than half of the vertical space between the cupboard and the tabletop. This causes the robot to have little room to maneuver when operating beneath the cupboard, putting the robot at a high risk of colliding with the cupboard or knocking over other grocery items. In several cases, the built-in planner was unable to plan a path to a location above a target item beneath the cupboard due to these restrictions.

Objects Near Environment Boundary

Early testing within the simulation environment revealed that if the cylindrical objects were knocked over at any point, there was a high risk that they would roll off the edge of the table. Even the cuboid objects, which would not move of their own accord, could end up getting pushed outside of the workspace incidentally. To mitigate this, the task space was edited to include a low wall around the edges of the spawning area. However, the end effector would

occasionally push an object up against the wall, and the resultant forces would prevent the robot from reaching its target position. It often became impossible to grasp objects in this situation.

Unreliable Simulation Behavior

The simulation caused several major challenges for this project. While the simulator provided a nice test bed to develop the learning and resetting agents, several of the behaviors will not transfer to the real world. The gripper in the simulator uses a “sticky finger” mechanism to determine if the grasp is successful. This method creates a successful grasp if the grasp command is called and any part of the gripper is touching the object. Resulting grasps are often not in the gripper and create a poor training example to the placement agent.

Another challenge with the simulator was unreliable behavior in executing actions. Often, the simulator simply did not execute commanded actions. This caused the system to intermittently skip tasks by the manual agent that were crucial to properly setting up the environment for the next episode in RL training. As a result, the RL agent would learn from improper scenarios that inhibited appropriate task training.

Methods Implemented

This project utilizes a hybrid systems approach to placing the items in the cupboard, and resetting the items to the table. Learning agents attempt to place the grocery items into the cupboard. After each learning episode, a manual agent returns the items from the cupboard to the table in preparation for another learning episode. Both types of agents operate in an eight valued action space corresponding to the absolute position (X, Y, Z), attitude as a quaternion (X, Y, Z, W), and gripper state.

RL Behaviors

The reinforcement learning (RL) for this project uses the library TensorForce. The TensorForce library is beneficial relative to other libraries because it can be set up agnostic to the training environment, meaning it does not rely on the reward and terminal conditions from RLBench. This enables modular reward development and seamless transition to real world training. This project uses the Deep Q Network (DQN) methods within TensorForce for the forward agent. Both agents operate in a 16 value input space consisting of the poses of the gripper and target, an integer representing the target, and the gripper state.

As previously mentioned, the grasping agent utilizes a DQN to map inputs to outputs. The agent makes two assumptions to simplify the action space. The first assumption is that the end effector is directly over top of the targeted item and does not command the X or Y position of the end effector. The second assumption is that an appropriate grasp orientation can be achieved by adjusting only the yaw of the gripper. With these two assumptions, the action space simplifies to Z position, yaw, and gripper actuation. The agent is rewarded for moving towards the target, and grasping the target. The agent is penalized for closing the gripper without grasping the target, knocking the target over, and moving away from the target.

The placement agent operates under the main assumption that the grocery item is properly grasped. As such, all orientation related learning is off loaded to the grasping agent and simplifies the placement action space to four values: cartesian position (X, Y, Z), and gripper state. The inputs to the placement agent are slightly modified from the grasping agent by setting the target pose to the middle of the cupboard, and the target number to the item being grasped.

Resetting Behaviors

To develop and test the resetting behavior, a manual agent was used to place groceries in the cupboard. This was accomplished in eight action steps. First, the robot stages itself above the desired object, then takes many small steps towards the object. When the robot detects contact, the grippers grasp the object. Next, the robot positions itself in front of the cabinet with an orientation that points the object into the cupboard. The robot then moves the object into one of four predetermined locations in the cabinet. The robot releases the object and moves back in front of the cupboard. Finally, the robot returns to the start position and repeats the cycle four times. Finally, upon placing the fourth object into the cabinet, the robot calls the reset function.

Testing for contact with the objects makes the reset function more robust. The robot tests for contact in three trouble spots: grasping objects on the table, grasping objects in the cabinet, and while randomly placing an object on the table. In each case, contact was a trigger to end the current action, and move on to the next action. Testing for contact was implemented by comparing the force on each joint after moving a small step. If the sum of the absolute differences of each joint was greater than 50 newtons, contact was made.

Both the manual forward task grasp and the cabinet reset grasp use the same approach. The robot would align the gripper orientation with the workspace and cabinet respectively and approach the object from only the Z direction, without rotating the gripper head. Once the gripper reaches the object, the forward task grasp closes the grippers to simulate the RL agent. The resetting grasp contacts the object with the gripper closed to simulate a vacuum grasp.

Once the robot had successfully retrieved an object from the cupboard, that object needed to be returned to the table to begin a new training episode. The system randomly selects an X and Y coordinate from within the task area and then determines a Z position based on the height of the object being held. The robot then moves the object to this location and releases the object. This process repeats until all items are redistributed across the tabletop workspace.

Evaluation

RL Behaviors

The grasping agent trained for approximately 30 hours and successfully grasps an object in 25-45% of training episodes. The large variation in success comes from two major sources, excluding insufficient training time. The first source of variation is inconsistent training examples. Throughout the training the environment was slightly modified by adding a bounding fence on the table so the objects do not fall to the floor. Additionally, as the arm was training it learned the undesired behavior of pushing the objects down before attempting to grab them. To combat this,

the reward function was modified throughout training to achieve more desirable behavior. The second inconsistency came from the inherent difference in difficulty between grabbing different objects.

The placement agent successfully placed the grocery item in the cupboard in 15-30% of training episodes. Training for this agent consisted of only 10 hours, which is likely the primary factor of low performance. Another major source of low performance comes from the difference between the training and evaluation scenarios. To accelerate training, a manual agent grasped the item at an ideal grasp point before staging the object for placement. In evaluation, the placement agent acted following a successful grasp from the grasping agent. Many of these learned grasps were unrealistic due to the “sticky finger” mechanism, and made placing the item much more difficult than any of the training episodes. A potential remedy for this would be to include a grasp evaluation metric for the final reward to the grasping agent. Recordings of the performance display these challenges[1].

Resetting Behaviors

The manual putting in the cabinet behavior was only needed to place more than one object into the cabinet successfully. From that viewpoint, the behavior was quite successful. The behavior, on average, put 3 out of 4 objects into the cabinet and kept them there. Any errors that we might have, such as sometimes having no objects in the cabinet, pushing objects very far into the back, and placing objects near walls helped the resetting algorithm to become more robust.

The testing for contact behavior was very beneficial in making the code more robust. While it would not detect contact by touching a sliding object, it would detect the arm hitting walls and objects that could not move. This allowed the arm to stop the action and try something else instead of endlessly attempting to grab an object that was behind a wall. This algorithm worked in the majority of cases the arm contacted an immovable object.

This straight grasp was sufficient in the majority of cases, but lacked the ability to capture edge cases. Some of those edge cases include strawberry jello on its side in the cabinet, objects very close to walls and deep beneath the contact point. As these are edge cases, there are some runs where the straight grasp gets the robot all the way through. However there is more work and grasp strategies needed to robustly reset the simulation everytime.

To validate the table resetting, the robot would select a location to place the item, and would attempt to reach that location within 25 actions. If it was unable to reach that location within the specified number of steps, it would simply drop the item at its current location. Over 50 repetitions of this behavior, the robot deposited the selected item in an acceptable location all but one time. In this one failure, interference from another item on the tabletop caused the robot to deposit the item outside of the boundary wall.

[1] https://drive.google.com/drive/folders/1B-rNnm6iG-XsnbDhMj_-5rxGnEFmsYoG?usp=sharing

Future Work

Implementing this model in the real world would require enhancements to the current project. In a real-life environment, the robot would not have pose information for the grocery objects. It would need cameras to assess the scene, locate the groceries, and determine their pose. Extra safety measures would also need to be enacted to ensure the robot does not collide forcefully with the cupboard or table. Work in this area would include a collision avoidance motion planner, and robust motion commands to the robot joint controller.

Additional work can be done to make the task more successful in real world application. Besides improving the RL functionality directly, the forward and resetting tasks can be slightly modified. An improvement on the forward task would be to designate unique locations for each grocery item in the cupboard. The resetting task currently only moves objects out of the cupboard and onto the table's surface; it does not place the objects on the table in any specific orientation. It would be beneficial to orient objects upright during the resetting task.