Real Robot Challenge Phase 2: Manipulating Objects using High-Level Coordination of Motion Primitives

Team: **Personal Robotics Lab**
Paul G. Allen School of Computer Science and Engineering
University of Washington

Rishabh Madan*
rishabhmadan96@gmail.com

Ethan K. Gordon*
ekgordon@cs.washington.edu

Advisor 1: Tapomayukh Bhattacharjee
tapo@cs.washington.edu

Advisor 2: Siddhartha S. Srinivasa
siddh@cs.washington.edu

November 13, 2020

Abstract

In Phase 1, we presented an approach to rigid-body manipulation based on carefully-designed motion primitives. We now demonstrate in Phase 2 that these primitives combined with a more robust state machine can reasonably complete all 4 tasks. For Phase 3, we propose using model-based and time-constrained (e.g. contextual bandit) RL to select and optimize primitives based on the environment. In general, our approach emphasizes the use of ML machinery only when classical approaches fail.

*These team members contributed equally to this work.

Phase 2: Real Robot

Jacobian Force Controllers and PID Loops As in Phase 1, we first reduced the more complicated joint space $\mathbf{q} \in \mathbb{R}^9$ (3 fingers $\times$ 3 joints per finger) into the more interpretable space $\mathbf{x} \in \mathbb{R}^9$ 3D Cartesian position of each end effector, discarding finger orientation due to the rotational near-symmetry of the end effectors. We retrieve the Jacobian matrix calculated from PyBullet: $\mathbf{J} := \frac{\partial \mathbf{x}}{\partial \mathbf{q}}(\mathbf{q}, \dot{\mathbf{q}})$. We can use its inverse to convert a task space force command into a joint torque command: $\dot{\mathbf{q}} = \mathbf{J}^{-1}\dot{\mathbf{x}}$. $\mathbf{J}$ may be singular or non-square, so we use its damped pseudo inverse to guarantee a good solution at the cost of slight bias: $\dot{\mathbf{q}} = \mathbf{J}^\top \left(\mathbf{J}\mathbf{J}^\top + \lambda \mathbf{I}\right)^{-1} \dot{\mathbf{x}}$. We combine this with gravity compensation torques from PyBullet's inverse dynamics module to command gravity-independent linear forces at each finger tip.

We build upon these linear force commands to create position-based motion primitives. Given a target position for each finger tip, we can construct a feedback controller with manually-tuned PID gains (see Figure 1). Since these systems are linear if we avoid collisions, multiple problems can be solved simultaneously through superposition.

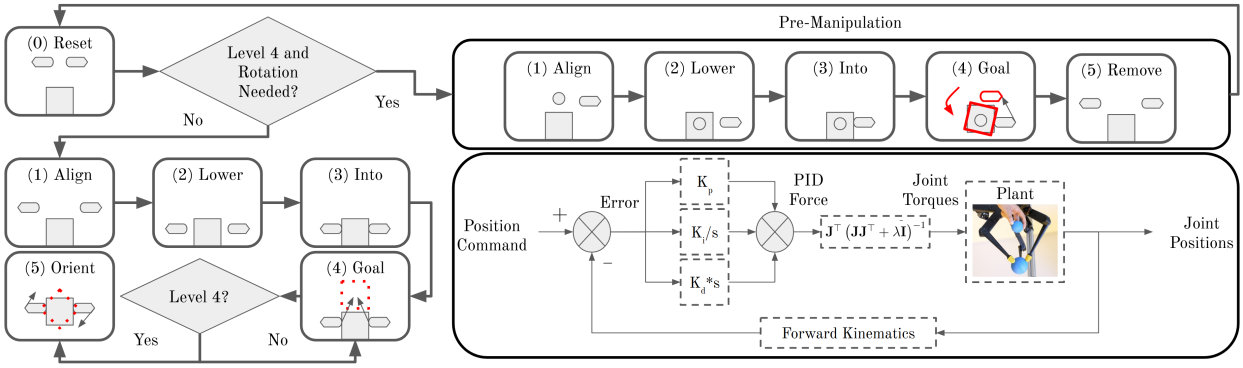

Figure 1: State Machines and Inner PID Loop. On interim failure (such as a time-out or dropped cube), all states will return to state 0 (*Reset*).

State Machine Finally, we combine a series of carefully-designed motion primitives into a simple finite state machine to reliably complete Levels 1-3.

0. *Reset:* Move arms to a pre-defined position well above the cube.
1. *Align:* Designate 3 contact points in an equilateral triangle around the cube. Move each finger to a point above these grip points (to avoid cube collisions).
2. *Lower:* Lower the fingers to the grip points.
3. *Into:* Initiate force closure by commanding a constant force towards the center of the cube at each finger until all finger tip force sensors are tripped.
4. *Goal:* Set the target pose to be the current pose plus the difference between the current cube location and the target cube location. Superimpose this onto the force from *(3)*.

In Levels 1-3, *Goal* continues indefinitely. If the robot spends too long in any given state, or the cube drops out of its fingers, it will return to *Reset* and retry.

Premanipulation for Level 4 For our submitted Level 4 code, we focus only on adjusting Roll and Pitch to minimize orientation error. Specifically, we implement an additional premanipulation state machine which flips the cube 90 degrees such that a new face is on top. The procedure is similar to [1], where two fingers target opposite sides of the cube, and the third facilitates the rotation. During *Reset*, we determine if the correct cube face is facing up (based on which face is most "on top" of the goal position). If not, we enter the premanipulation state machine a maximum of 2 times, rotating up to 180 degrees.

We handle small pitch/roll errors by adding *5 (Orient)*. Set the target force at each finger orthogonal to *(3)* in the direction of pitch/roll error. This imparts a torque on the cube in the desired direction. We superimpose this onto the forces from *Goal* and continue indefinitely.

Results and Discussion The above procedure performs fairly well on Levels 1-3. We sample 10 goal configurations for each difficulty level and run jobs on the different machines. In our final tests before submission, for Level 1, the average cumulative reward was -3131.37, Level 2 had -3641.90, and Level 3 had -3380.0.

For Level 4, the baseline approach without the premanipulation state machine had an average cumulative reward of -47337.2082. Adding the pitch correction, we did see an improvement to -32254.82. Unfortunately, we were unable at this time to implement the proposed second premanipulation state machine to add yaw rotations. Therefore, while the top face of the cube was reasonably aligned, the other faces were not. Our immediate next step will be to finish the implementation of this second state machine to further reduce the yaw difference.

Phase 3: Different Object Shapes

Our results were promising enough that our proposal for this phase has not changed. While we do not yet have the details on this phase, we can envision two possible paths here. If the object is more difficult, but does not change between trials, we could re-use the work from Phase 2 (with the addition of yaw correction). Designing motion primitives by choosing points on a 3D model of the object, using ICP with any RGBD sensors to align them with the real object.

Alternatively, if the shape changes regularly, we can consider having an online learning algorithm that can select the optimal motion primitive given information about the object. Previous work done by this team in the manipulation of deformable objects has shown promise using this contextual bandit / restricted RL approach [2].

Available Resources Now that October 31, 2020, has passed, we expect to be able to commit 10-15 person-hours per week with the exception of the week of Thanksgiving. While this approach has yet to require it, we do have access to multiple GPU clusters, including at minimum direct unrestricted access to a machine with 4 Nvidia 1080Ti GPUs.

Video Attachments Playlist

https://www.youtube.com/playlist?list=PLEYg4qhK8iUaXVb1ij18pVwnzeowMCpRc

1 References

[1] "In-hand turning," https://www.youtube.com/watch?v=xu5VvyjDLRY,[Online; Retrieved on 24th September, 2020].

[2] E. K. Gordon, X. Meng, M. Barnes, T. Bhattacharjee, and S. S. Srinivasa, "Adaptive robot-assisted feeding: An online learning framework for acquiring previously-unseen food items," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020.