Learning to Manipulate Objects using High-Level Coordination of Motion Primitives

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

Ethan K. Gordon
ekgordon@cs.washington.edu

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

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

September 25, 2020

Abstract

We present an approach to rigid-body manipulation based on the high-level optimization and coordination of carefully-designed motion primitives. In Phase 1, we combine these primitives into a simple state machine to complete all 4 tasks. In future phases, we propose the use of 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.

Phase 1: Simulation

Jacobian Force Controllers and PID Loops 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: $\mathbf{J} := \frac{\partial \mathbf{x}}{\partial \mathbf{q}}(\mathbf{q})$. 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 available 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.

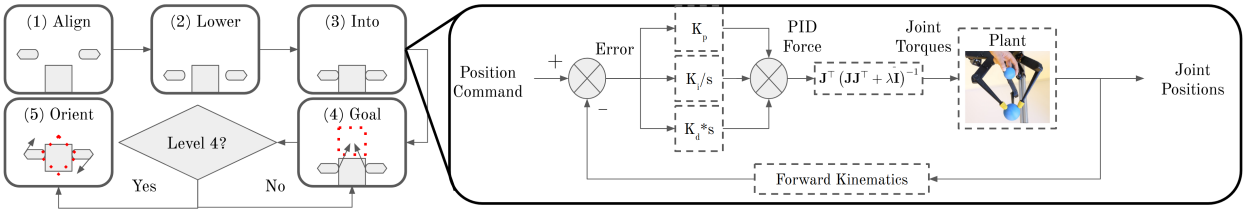

Figure 1: State Machine and Inner PID Loop

State Machine Finally, we combine a series of carefully-designed motion primitives into a simple finite state machine to reliably complete Levels 1-3. *(1)* 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 the fingers to the grip points. *(3)* 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)* 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, this state continues indefinitely.

Premanipulation for Level 4 We handle small orientation errors by adding *(5)*. Set the target force at each finger orthogonal to *(3)* in the direction of orientation error. This imparts a torque on the cube in the desired direction. Superimpose this onto the forces from *Goal* and continue indefinitely.

This does not work for large rotations. We therefore implement an additional in-hand premanipulation step which flips the cube 90 degrees to whichever orientation is closest to the target orientation. The procedure is similar to [1], where two fingers target opposite sides of the cube, and the third facilitates the rotation.

Results and Discussion The above procedure performs very well on Levels 1-3. For Level 1, the average last-step reward was -0.0158, Level 2 had -0.001, and Level 3 had -0.002. In effect, all three problems were solved near-perfectly by this metric, though the cube could definitely be moved more quickly, and Level 1 ran into some additional steady-state error due to ground friction. Both are issues that can be addressed by more intense PID tuning.

Level 4 ran into more trouble due to the limited time allowed for manipulation. We could execute only one 90 degree pitch or roll rotation, which did not allow time to handle large yaw rotations. With more development time, we could potentially speed up our primitives enough to allow for a second pre-manipulation task to remedy this. Our average last-step reward for Level 4 was -0.209.

Phase 2: Real Robot

The above approach worked well in simulation in part because we had perfect knowledge of the environment and noiseless actions. We would consider adding a Bayes filter to properly handle uncertainty in perception (and by extension state estimation). However, this additional lag vs. noise trade-off could make it much more difficult to design stable PID controllers that still complete the task quickly. If this is the case, we can consider systems like general MPC or iLQR [2], assuming that the environment model is still tractable. If not, we can likely use the simplicity of this sub-problem to try a model-based RL solution like MPPI [3], which has shown promise in other high-speed scenarios.

Phase 3: Different Object Shapes

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. 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 [4].

Phase 4: Writing

The main difference here is the importance of intermediate states while maintaining appropriate forces and positions in different directions. The pen must always touch the paper with the appropriate range of forces while maintaining the position trajectory for writing the character. This is a classic case of hybrid force-position or impedance control: controlling pen-paper and gripper-pen interaction forces while tracking the desired position for the character to be written. Additionally, we could consider adding a constrained motion planning algorithm that minimizes full-trajectory deviation, such as [5]. This would likely require an additional mechanism for anomaly detection and recovery during trajectory execution.

In conclusion, our philosophy is to approach each task with the simplest solution that still works, only adding in additional complexity when absolutely necessary.

Available Resources Prior to October 31, 2020, which is the majority of Phase 2, we can likely commit on the order of 5 person-hours per week of dedicated work, plus additional advising time. We expect to be able to add 5-10 more person-hours per week after that date. 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.

1 References

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

[2] W. Li and E. Todorov, "Iterative linear quadratic regulator design for nonlinear biological movement systems," *ICINCO (1)*, 2004.

[3] G. Williams, N. Wagener, B. Goldfain, P. Drews, J. M. Rehg, B. Boots, and E. A. Theodorou, "Information theoretic MPC for model-based reinforcement learning," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 1714–1721.

[4] 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.

[5] R. M. Holladay and S. S. Srinivasa, "Distance metrics and algorithms for task space path optimization," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2016, pp. 5533–5540.