6.4210 Final Proposal: Throwing and Catching Bot

Throwing and catching are very important topics in the field of robotic manipulation, as they are prerequisites for more complex robotic manipulations such as badminton, basketball, juggling, table tennis, and so on. Even in simple daily tasks, humans use throwing skills when putting something in trash, passing an object to a friend, or putting objects in boxes that are not in reach. Similarly, humans use catching skills while receiving an object from someone else or when a glass or plate falls from the side of a table. As a team of 3, we will be undergoing a project that explores the dynamics of catching and throwing objects between two separate robots in motion. The project is intended to be a multi-step process that will culminate in a simulated demonstration of these robots throwing and catching a simple object back and forth repeatedly in which the catching robot will have to move. If time permits we will attempt to replicate at least one successful throw and catch on real-world hardware.

More precisely, we can describe the problem as follows. First, there is the throwing problem. Questions such as planning the throw trajectory, being able to implement effective control for differently shaped objects, and how to move to and plan a reasonable position to maximize the success of the throw are some of the major questions that are present in just the problem of throwing. Then of course there is the catching problem. For this operation, there is the challenging problem of completing real-time trajectory planning of perceived aerial objects in a very short time. At the same time, achieving a good prediction for object trajectories, and optimizing the capture mechanism of the end-effector are both interesting problems that will prove challenging, and have many applications outside of direct throwing and catching, such as safe human-robot interaction under uncertainty.

Some important topics covered in class that we will be applying to our project include geometric perception, determining optimal grasp positions, motion planning. Geometric perception is an important concept for our problem because we need to determine two things about the object: its pose in 3d space and the location of the center of mass of the object with respect to the object frame. The center of mass of the object is an important aspect of our task because when the grasper releases the object, the center of mass will follow a particular trajectory but the rest of the object might fluctuate. We will be implementing what we learned about optimal grasping since as opposed to the pick and place problem, we might need to impose additional constraints on grasping to adapt to a throwing problem. For instance, to not generate too much torque on the object, it is best if our antipodal grasp is near the center of mass. Lastly, a major topic covered in class that will be pivotal in achieving our task is motion planning. While throwing an object, we need to first determine a point for releasing the object and the velocity at which it must be released. Then, we need to plan a trajectory for the robot such that the robot reaches the desired velocity at the given point under acceleration constraints. The acceleration constraints arise because there is an upper bound on how much static friction there can be between the grasper and the object before the object starts slipping. This is a motion planning problem unique to this instance because we do not have a particular time when we want to throw the object. The information and experience we've gathered in class will be a solid starting point for tackling several of the problems present in our project.

Building on the foundational knowledge gained in class, we will be incorporating some previous research work to help us achieve our project goal. We will be referencing "**A Solution to Adaptive Mobile Manipulator Throwing**" *(Yang Liu, et al.) as* this paper introduces mobile

manipulator throwing. This paper can be useful for the team to learn how to implement throwing and apply it to the case of mobile manipulators. We will also be looking at “**Revisiting Ball Catching with a Mobile Manipulator: A Discrete Trajectory Planning Approach**” *(Ke Dong, et al.)* which can help us to understand how to catch a ball with a mobile manipulator. This paper addresses this problem by constantly interweaving predictions and replanning to successfully catch the ball in the air and implementing a new and different approach to the problem than Sequential Quadratic Programming (SQP). Another important paper that we feel will be critical to our work is "**TossingBot: Learning to Throw Arbitrary Objects with Residual Physics**" *(Andy Zeng, et al.)* which will help us in understanding the challenges and methods of throwing an object. Lastly, we will be referencing "**Coordinated multi-arm motion planning: Reaching for moving objects in the face of uncertainty**" *(Mirrazavi Sina, et al.)* since this paper might give us some idea about catching an object in motion and under uncertainty since the paper involves intercepting a moving object. The paper also has an example where they catch a fast-moving object with the coordinated efforts of two robots: a task we may consider exploring for demonstration purposes.

To be successful and to make steady progress in our project, we will assign each member individual tasks and convene bi-weekly to discuss progress, and if need be, integrate work. By **Nov 9 (Check-in 1)** we will have had the simulation environment set up, registered necessary scene objects, and implemented basic geometric perception and grasp selection for the throwing robot. By **Nov 30 (Check-in 2),** we will have solved both the problem of creating throw trajectories and motion planning for throwing to a fixed location. For the second robot, we will combine the geometric perception algorithms implemented earlier with trajectory prediction to generate an interception trajectory, for instance, by using the dynamical system methods described by authors Mirrazavi Sina, et. al. Lastly, by **Dec 7** we will use the throwing algorithm and optimize the throw location so that the second robot can catch the object. As time allows, we aim to test for different configurations of the robots and different (but known) objects. By the end of the project period, we hope to fully realize the basic tasks of two robots throwing and catching objects through the content learned in class and consulting a lot of extracurricular materials. It's going to be tough, but we love a challenge.

6.4210 Final Proposal: Throwing and Catching Bot

High Level Description	**More detailed description**	**Code**	**Priority/date**
Grasp the brick **-Quincy**	Grasp the brick at the center of mass	Input: robot position and brick position Output: catch the center of the brick and go to the initial position	
Trajectory design/Motion planning **Hanqi**	Throwing trajectory design	Input: Current pose of the robot, final pose, or maybe even time. Maybe constraints? Output: Poses for the robot (end effector) at particular times.	
Inverse kinematics/dynamics, whatever we use **Arif**	Given a trajectory for the robot, determine the joint angles/forces depending on which type of control we are going to use	Input: A trajectory (list of poses, times) Output: list of robot joint angles.	
Using the low level controller for the robot **-Arif**	The inputs to the robot (forces or any other control method, joint velocities)	Will figure out how to command the robot in drake	
Perception for red brick (catching) -**Quincy**	The catching robot needs to estimate the trajectory of the brick, so we need the pose of the brick at certain times.	Input: camera data Output: Bounding box of brick, pose of brick in world frame	We might be able to get the position of the red brick from the simulation. If we can do so, we can delay doing this.
Predicting future poses of the brick **Hanqi**	Given the history of poses for the brick, we need to predict the pose at some time t in the future. We could use polynomial fitting (position should be a quadratic of time) or	Input: history of poses, times Output: the next k poses	Important for catching

6.4210 Final Proposal: Throwing and Catching Bot

	physics based methods for this,		
Create intercept trajectory for catching -**Arif**	Design intercept trajectory given some intercept point within the robot's operation space.	Input: Output:	
TBD 1)maybe catch a small box to receive the brick so that help to achieve the catching problem 2)and pull down the brick into the larger box 3)and once again to catch the brick to throw			