EE 106A Final Project Proposal
Ball Catching Robot with MPC and Dynamic Sensing
Fall 2017

1. Contact Information

Name	SID	Email	Experience
Philipp Wu	26303669	phil80301@berkeley.edu	(Undergraduate) Electrical Engineering and Computer Science and Mechanical Engineering, Pieter Abbeel's Robot Learning Lab, JPL, Space Technologies at Cal
Rachel Thomasson	25604955	rthomasson23@berkeley.edu	(Undergraduate) Mechanical Engineering, Pieter Abbeel's Robot Learning Lab, EnableTech DeCal Facilitator
Albert Li	26661255	alberthli@berkeley.edu	(Undergraduate) Mechanical Engineering, Minor in EECS, Ken Goldberg's AutoLab, RoboBears, Space Technologies at Cal - Rover Design Team
Kireet Agrawal	26379648	kireet@berkeley.edu	(Undergraduate) Electrical Engineering and Computer Science GoDaddy Inc., MyReviewers LLC, Space Technologies at Cal - High Altitude Balloon Team
David Gealy	22747689	dgealy@berkeley.edu	(Graduate) Mechanical Engineering PhD student Pieter Abbeel's Robot Learning Lab, Ken Goldberg's AutoLab, North American Repower

2. Abstract

Building a 7 DOF robot arm (extension of research), developing Model Predictive Controller and a Computer Vision system for the purpose of catching some sort of object in real time. The object would be something large and trackable like a balloon or beach ball which can give the robot some time to react. The robot will catch the object by having some velcro attached to the end effector to help the object stick to the robot. We abstract this final goal with:

3. Project Description

Goals**:**

- Create a real time MPC controller for a robotic manipulator such that it can plan an optimal trajectory to catch(or touch) a ball. Our robot will also have dynamic sensing of the ball, with real time tracking and prediction of the ball's location. Our project then must have a controller to go to a position to reach the ball, as well as have the ability to find and track a ball with computer vision.

Challenges:

- The problem of having a very short time horizon while the ball is in the air makes the problem difficult, and constrained. This lends itself well to a MPC and finite time optimization control problem but must have the ability to compute quickly and allow the arm to reach the ball in time. There must be dynamic sensing and replanning as better estimates of the ball's trajectory are determined.

Sensing:

- For the Sensing component we will use image processing and computer vision to track the trajectory of the ball(flying object). This will be done in real-time through Kalman Filtering and multi-sensor fusion, using cameras and a Kinect sensor.

Planning

- This problem has a finite time in which the robot must achieve its task otherwise the object will fall to the ground. This lends itself well to a MPC problem as we have a known and limited actuation with a finite time horizon over which we can actuate over. Here we plan to have the robot estimate the eventual location of the ball, and compute the optimal trajectory to reach and catch the ball.

Actuation

- After calculating the trajectory, of course, the manipulator will actuate according to the desired inputs. Hopefully reaching the desired pose and catching the ball.

Similar Work

- https://actu.epfl.ch/news/super-fast-robot-arm-can-catch-whatever-you-throw-/
- http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8062503&tag=1
- Similar research for catch objects with a robot arm exist, as well as using MPC methods to control arms. However we are using an arm not designed specifically for this task, to catch an object, using optimal control methods. Furthermore we are exploring different CV methodologies such using a kinect sensor fused with webcams to accurately track our object.

4. Tasks

1. **Build the Robot.** We will build a seven degree-of-freedom, modular, cable-differential driven robot arm.
2. **Develop Control Structure for the Arm.** We will design and program the control architecture for the arm from the high level trajectory planning down to the low level position controllers(which send the direct current commands to the motors)
    a. Creating a robot model for simulation and control testing purposes.
    b. Creating a Gazebo simulation environment for the robot to test control code.
    c. Determining the specific MPC architecture to use in arm trajectory planning.
    d. Optimize control code and test for computational feasibility given the short time horizon.
    e. Tuning parameters such as objective function, receding horizon length as well as improving the robot model and constraints.
    f. Once Tasks 4 and 5 are completed, develop ROS node to publish inputted T-timesteps to ball_input_timesteps ROS topic in order to predict the ball's location, and subscribe to the ROS topic expected_ball_location to obtain a real time estimate for the location of the ball.
    g. Develop ROS node that uses the above method to control arm to the given estimated ball location.
3. **Perform Tests to Determine Ball Choice**
    a. We will perform simple tests on multiple types of balls (e.g. whiffle ball, tennis ball, velcro ball, ping pong ball)
    b. Using qualitative observations, we will select the ball that maximizes time in the air without being significantly affected by drag forces.
4. **Develop Ball Sensing Method.** We will using image processing, computer vision and sensor fusion to sense the thrown ball
    a. For the video camera sensing method, we will use a real-time bit-masking algorithm written in Python using the OpenCV library to detect distinctly colored balls in real time.
    b. We will also use Hough Circle Filtering and Canny Edge detection as another method for video camera sensing method, written in Python and using OpenCV.
    c. Using the Kinect we will utilize the Kinect's depth sensor to detect ball motion.
    d. Aggregate all predicted ball locations from the multiple sensors, using sensor fusion with Kalman filtering (http://www.ijfis.org/journal/view.html?uid=785&&vmd=Full) and weighted linear regression on distances.
    e. Develop ROS Node to aggregate sensor data, process camera and Kinect data, and publish the ball's predicted current timestep location in the Robot arm's world frame to a ROS topic, ball_current_location.
5. **Develop Ball Trajectory Planning.** We will find a predicted location for the ball in the Robot arm's world frame.
    a. In the robot arm's world frame, we will use parabolic trajectory planning to estimate the ball's location after an inputted T timesteps, written in Python or C++.
    b. Develop ROS node to subscribe to the ball_current_location ROS topic and ball_input_timesteps ROS topic, aggregate the known current path in memory and utilize

the trajectory planning function written above with an inputted expected time T to estimate the ball's location for, and the publish to a ROS topic, expected_ball_location.

5. Milestones

- Being able to accurately estimate the trajectory of a ball (and its position when close to the end effector of the arm) thrown through the air using multiple sensors (Webcams, Kinect) and Computer Vision (Saturday, 10/28/17)
- Developing a dynamic model for the arm (Saturday, 10/28/17)
- Implementing a path-generation algorithm like RRT based on estimated contact point with ball and initial state of the arm (Sunday, 11/5/17)
- Developing optimal controller for trajectory following for the arm (Sunday, 11/19/17)
- Simulating the response of the arm in modeling software to verify that our control works and is robust to injected noise (Sunday, 12/3/17)
- Testing the controller in real time on a physical arm and tuning gains (Sunday, 12/10/17)

6. Assessment

- In order for this project to be considered a success, the end effector of our custom arm will have to coincide with the ball at some point along its path.
- Realistically, the arm will move so that the end effector simply touches the ball.
- An idealistic goal is being able to catch the ball, either by fixing a net/basket to the end effector or by making use of a velcro pad that and velcro strips attached to the ball used.
- Considering that complications may arise in building the custom 7 DOF arm, using a Sawyer to catch the ball will also be considered a success, however in this case we will require the robot to be able to catch the ball rather than simply touching it.

7. Team Member Roles

- Philipp, Rachel and David will collaborate on building and developing the custom 7 Degree of Freedom robot arm. (Task 1)
- Philipp, Rachel and Albert will work on developing MPC and motion planning for the arm. (Task 2)
- Rachel and Albert will work on testing multiple balls to choose the optimal ball that is well estimated by a parabolic trajectory planning method. (Task 3)
- Kireet will work on sensing using multiple sensors and real time vision algorithms. (Task 4)
- Philipp and Kireet will work on the ball trajectory planning method. (Task 5)

8. Bill of Materials

EE106a Lab Resources

Item	Quantity
Sawyer (for testing vision methods)	1

Other Items

Item	Quantity	Owner/Location
Xbox Kinect	1	RLL/444 Soda
7 DOF Custom Robot Arm	1	RLL/444 Soda
Ball (Type TBD)	1	RLL/444 Soda
Velcro Mitt (Potentially)	1	RLL/444 Soda
Web Camera	1	RLL/444 Soda

9. Other

Preliminary Code Structure

Main Project Repository: https://github.com/phil80301/ee106a
Sensing: https://github.com/kagrawal2/Robot-Arm-Vision-Tracking

ROS Packages for Control

http://wiki.ros.org/tf
http://wiki.ros.org/kdl

Sensing Libraries

OpenCV: https://opencv.org
Sk-Learn: http://scikit-learn.org/stable/