Robotic Manipulation 2015

Project 2 (4752/4750: 100 points, 5752/5750: 130 points)
Obstacle Avoidance and Contact

Due at the start of class, Wednesday, Nov. 4, 2015

Objective: To demonstrate techniques for obstacle avoidance via motion planning and obstacle contact via differential kinematics.

Instructions: This is a group exercise. Turn in to CMS one compressed file (zip or tarball) for each group of three students. Include:

- your ROS package (be sure to do a "make clean" in your package before submission to minimize the size.
- a README file with instructions on how to run your code.

Write the name of your group and the names of the members at the top of each of your files. In addition, **you will demo your code to the instructors** at a time to be scheduled after the deadline. We will target questions to each member of the team to ensure equal contributions. If you want help finding a team where you can contribute more equally, please contact us.

Assignment: Complete the following assigned problems. Students signed up for the 5000-level version of the course may pick any combination of "extra credit" options to reach the full 130 points. Be warned that this assignment is a lot of work. Start early!

I. (50+30 points) **Line-following in Cartesian space.** You will write a program to make Baxter write on a whiteboard.

   1. (15 points) In Project 1, you used position control to move the arm from configuration q_1 to q_2 along a straight line in configuration space (C-space) at a constant velocity $\dot{q}$. In Project 2, you will instead use velocity control to make the end-effector move along a straight line at constant velocity ξ in the workspace from point to point. Use the Jacobian to implement simple line following in the workspace given two arm configurations end-effector positions (x_1 and x_2) in software by solving the equation $\xi = J\dot{q}$. Select a fixed workspace velocity, and iteratively use the Jacobian to command an instantaneous C-space velocity $\dot{q}$ via velocity control. The end-effector should stop when it reaches the goal position. Note that any configuration that attains the correct final end-effector position x_2 is acceptable, even if it is different from q_2.
   2. (10 points) Implement a simple proportional controller to correct for accumulating position error. To do this, you will need to keep track of time using `rospy.Time`. Note the time that the trajectory starts executing. Coupled with the desired velocity, compute where the end-effector should be at each iteration. Compute the difference between where it is and

where it should be. Add this error term as a second linear velocity vector. Scale the error by some constant $\kappa_p < 1$ to avoid overshooting the desired trajectory. When you use the Jacobian to command resulting C-space velocities, you are creating a simple closed-loop proportional controller. Once you tune κ_p, the hand should more accurately follow the line.

3. (10 points) Define a plane, for example by touching the end-effector to three points in the plane. Write a program that consecutively follows lines in this plane. Once it is working, put a dry-erase marker in Baxter's hand and write/draw something interesting on the whiteboard that is resting on the table or standing vertically next to Baxter. Be aware of the difference between the end-effector velocity and the tool velocity (Section 4.7 in the textbook Spong, Hutchinson, and Vidyasagar). HINT: writing on the whiteboard with Baxter in velocity control mode will probably be easier with the pen at about a 45° angle to the board instead of 90°. However, the exact angle of the pen is less important than the position of the tip.
4. (10 points) When using the Jacobian pseudoinverse to solve for ξ, use a secondary objective function to ensure that Baxter is well-behaved within the whole workspace. He should not hit singularities or joint limits when it is avoidable.
5. (15 points – extra credit) Implement code to follow splines (smooth curves) in addition to straight lines.
6. (15 points – extra credit) Turn Baxter into a typewriter. When a key is pressed on the keyboard, Baxter should write the corresponding letter on the whiteboard.

II. (50 + 30 points) **Motion planner in configuration space.** Implement a Rapidly-exploring Random Tree (RRT) motion planner. For reference, see: Kuffner and LaValle, "RRT-Connect: An Efficient Approach to Single-Query Path Planning"

1. (5 points) Construct a sampler that randomly selects points within Baxter's reachable C-space (i.e. within joint limits).
2. (15 points) Build a database in which to store good samples. Pick a metric on points in C-space. You should be able to query the nearest neighbor of an arbitrary point in C-space and have it return the nearest point already in the database. HINT: use a library like scipy.spatial.KDTree in Python.
3. (10 points) Write a routine to solve the boundary value problem (BVP). That is, given two samples in the configuration space, connect them by a path segment. You can connect them any way you like, but you should be able to defend your choice.
4. (10 points) Collision-check points and path segments using a library we provide. Return true of the geometry of the robot would collide with some obstacle while in the configuration(s) specified by the point or path.
5. (10 points) Write a routine to execute the planned path on the robot using velocity control.
6. (15 points – extra credit) Implement smoothing by removing unnecessary segments of the plan before executing it. This can be accomplished by a randomized algorithm that selects two points on different segments of the path and tries to connect them with a straight line.
7. (15 points – extra credit) Implement a bi-directional RRT.

Demo: Each group will demonstrate their code to the teaching staff and describe how it works. Your group will demonstrate drawing on the whiteboard and using motion planning to avoid colliding with obstacles. You can combine them into a single task or separate them out. If you impress the teaching staff by doing something technically interesting, that may be worth extra points.