CS4610/CS5335: Homework 1

Out: 1/27/16, Due: 2/5/16

Please turn in this homework to Rob Platt via email on the due date. HW Q1 and Q2 should be submitted as a PDF. HW PA Q1-Q5 should be submitted in the form of a set of five files named Q1.m ... Q5.m. All this should be zipped up into a single file and emailed to me.

Have a look at the accompanying zip file. Stub files for Q1.m ... Q5.m are provided to you. You should implement each of these. Once implemented, you should be able to run "hw1(X)" in order to run code for question "X". hw1.m is given to you and should not need to be modified. The only thing you need to do is to insert code into the stub functions in Q1.m .. Q5.m.

HW Q1: (Spong, Problem 2-15) If the coordinate frame A is obtained from the coordinate frame B by a rotation of $\pi/2$ about the x-axis followed by a rotation of $\pi/2$ about the fixed y-axis, find the rotation matrix R representing the composite transformation. Sketch the initial and final frames.

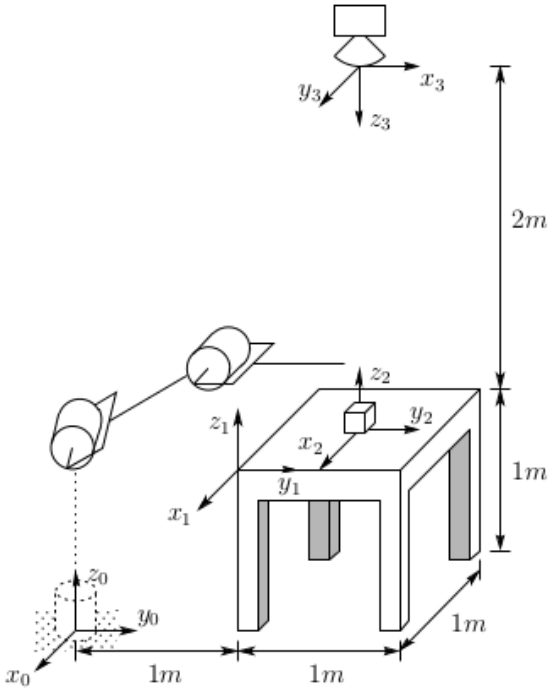
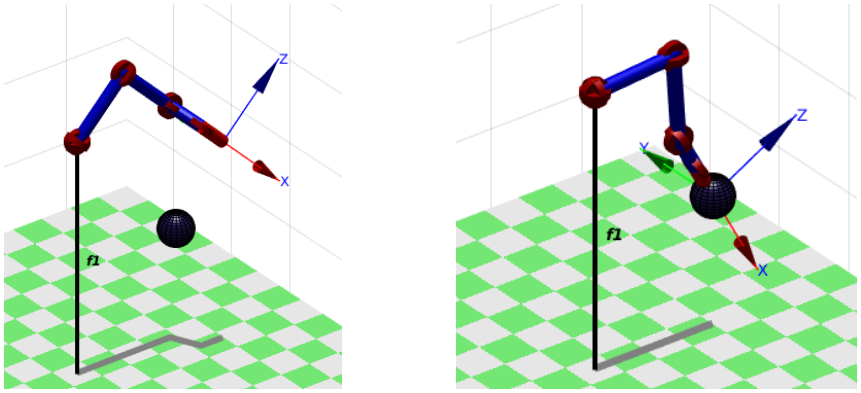


Figure 1:

HW Q2: (Spong, Problem 2-37) Consider the diagram in Figure 1. A robot is set up 1 meter from a table. The table top is 1 meter high and 1 meter square. A frame $o_1\ x_1, y_1, z_1$ is fixed to the edge of the table as shown. A cube measuring 20 cm on a side is placed in the center of the table with frame $o_2\ x_2, y_2, z_2$ established at the center of the cube as shown. A camera is situated directly above the center of the block 2m above the table top with frame $o_3\ x_3, y_3, z_3$ attached as shown. Find the homogeneous transformations relating each of these frames to the base frame $o_0\ x_0, y_0, z_0$. Find the homogeneous transformation relating the frame $o_2\ x_2, y_2, z_2$ to the camera frame $o_3\ x_3, y_3, z_3$.



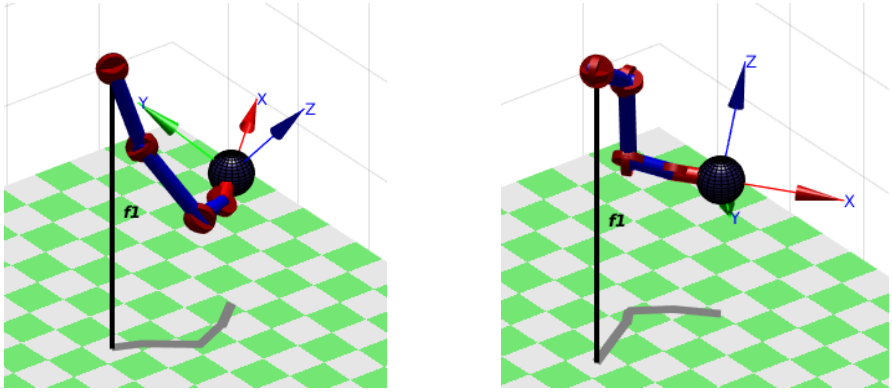
(a) (b)

Figure 2: Illustration of Q1. (a) is before moving the arm. (b) is after moving the arm to the configuration calculated in your function.

PA Q1: Implement the function in Q1.m. This function will use the built-in inverse kinematics function in the RTB to calculate a joint configuration that corresponds to a desired end effector position (just position, not orientation). The function will take as input a robot (encoded as a SerialLink class) and a desired position (encoded as a 3x1 vector). It will calculate a target joint configuration that will cause the end of the robot arm to reach a point at the center of the sphere (see Figure 2). This function should work for arbitrary desired positions.

PA Q2: Achieve the same result as in PA Q1, but this time using Jacobian pseudoinverse control. The exact solution found by your function will probably be different from what you found in PA Q1. However, the end effector should reach the same goal positions (the solution found by my code is shown in Figure 5(a)).

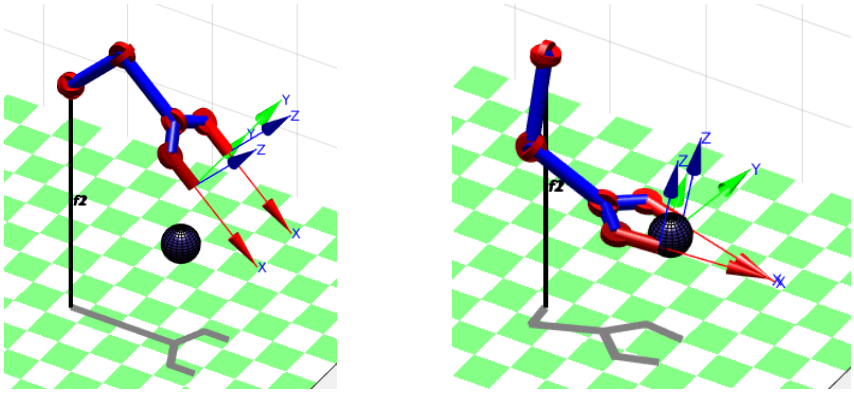
PA Q3: Achieve the same result as in PA Q2, but this time adding a Nullspace term that tries to keep the arm as close as possible to the initial configuration. The solution found by my code is shown in Figure 5(b)).



(a) (b)

Figure 3: One possible arm configuration found after running the code for (a) Q2 and (b) Q3.

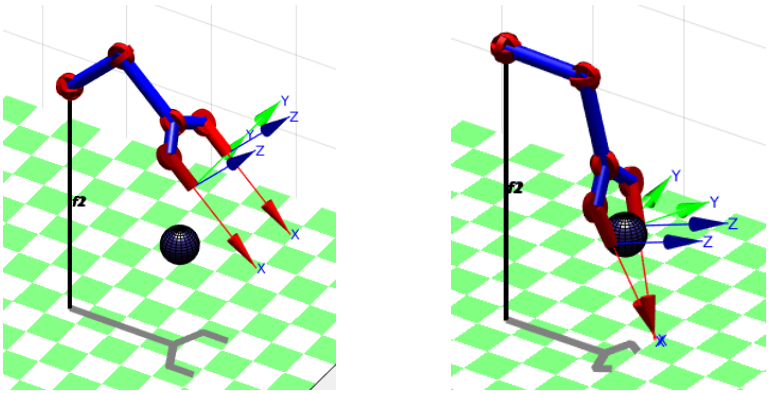
PA Q4: Now, imagine that there is a two-fingered hand on the end of the arm. Use Jacobian pseudoinverse control to move the arm/hand so that the two fingers capture the sphere by moving each finger to one side of the sphere as shown in hw1.m. This is a challenging problem. There are now TWO objectives in this problem – to move each finger to the desired spot. You will need to formulate a new Jacobian matrix that reflects this two-part objective. Notice that the configuration fo the arm and two fingers is now encoded as an 11-dof (degree of freedom) configuration rather than a 9-dof configuration. The first seven joints are the arm joints. The next two joints are for finger f1. The final two joints are for finger f2. You should use a step size of no more than 0.05 in this question.



(a) (b)

Figure 4: Joint configurations found by my code before (a) and after (b) running the code in Q4.

PA Q5: Notice that it is hard to get the code in Q4 to take a "direct" path to the goal. In this question, use a Nullspace term to cause the arm to take a shorter path. The Nullspace term should give the arm a preference for configurations that are close to the initial configuration (just as in Q3).



(a) (b)

Figure 5: Joint configurations found by my code before (a) and after (b) running the code in Q5.