ECE4560 - Solution #11

Problem 1. (40 pts) Consider, as usual, the three-link rotational planar manipulator from the previous homeworks, c.f. Figure 2. We have examined it a bit early on in the semester. Let's start to explore more the geometry of its movement, but including the rotation part. $SE(2)$ is nice in that the orientation is given by a single variable, which means that we can pretend that $SE(2)$ is vector-like and then just use the calculus we know (thereby avoiding body and spatial coordinates). This homework problem explores that aspect before we move to the body form of things. Here, you will map velocities in joint space to velocities in the configuration or end-effector space (sometimes configuration space is called the *task space* since all tasks are in terms of group configurations).

Let the link lengths be $l_1 = 1$, $l_2 = \frac{1}{2}$, and $l_3 = \frac{1}{4}$. Further, consider that the joint angles are set to $\alpha = (\pi/3, \pi/4, \pi/12)^T$ and the instantaneous joint velocities are $\dot{\alpha} = (\pi/12, \pi/8, 0)^T$.

(a) Now, let's consider the vector form of the forward kinematics. It will be $(x_e(\alpha), y_e(\alpha), \theta_e(\alpha))^T$. The only difference between this and the earlier version from the homework is the additional orientation coordinate. What is the manipulator Jacobian for this vector function?

(b) For the setup (e.g., the α and $\dot{\alpha}$) described, what is the vector form of the end-effector velocity for the given joint configuration and joint velocity? Will the manipulator rotate clockwise or counter-clockwise? (remember the right-hand rule versus clock direction)

(c) To get a feel for the velocity, plot the manipulator for the specified joint confiuration, then use `velocityPlot` function to plot it. It will plot the linear velocity correctly and it will plot the angular part in the first quadrant if the angular velocity is positive, and in the fourth quadrant if negative. The size does not scale with the angular velocity. This will visualize where the manipulator will move given the joint velocities and in which direction it will rotate. Turn in the plot.

The function `velocityPlot` is available as a homework file and should be added to your $SE(2)$. Actually there will be another useful function in the file called `twistPlot` which you will need later. Make sure to copy both in. They are complete and work.

Code: functions made available as plotCode m-file, to be integrated into your $SE(2)$ class as member functions.

Solution 1.

(a) As noted, all that is needed is to add the orientation coordinate to the earlier position-only forward kinematics, as per:

$$\vec{g}_e(\alpha) = \left\{ \begin{array}{c} l_1 \cos(\alpha_1) + l_2 \cos(\alpha_1 + \alpha_2) + l_3 \cos(\alpha_1 + \alpha_2 + \alpha_3) \\ l_1 \sin(\alpha_1) + l_2 \sin(\alpha_1 + \alpha_2) + l_3 \sin(\alpha_1 + \alpha_2 + \alpha_3) \\ \hline \alpha_1 + \alpha_2 + \alpha_3 \end{array} \right\}.$$

Computing the Jacobian, we get

$$J(\alpha) = \frac{\partial \vec{g}_e}{\partial \alpha}(\alpha) = \left[\begin{array}{cc} -l_1 \sin(\alpha_1) - l_2 \sin(\alpha_1 + \alpha_2) - l_3 \sin(\alpha_1 + \alpha_2 + \alpha_3) & \cdots \\ l_1 \cos(\alpha_1) + l_2 \cos(\alpha_1 + \alpha_2) + l_3 \cos(\alpha_1 + \alpha_2 + \alpha_3) & \cdots \\ \hline 1 & \cdots \end{array} \right.$$

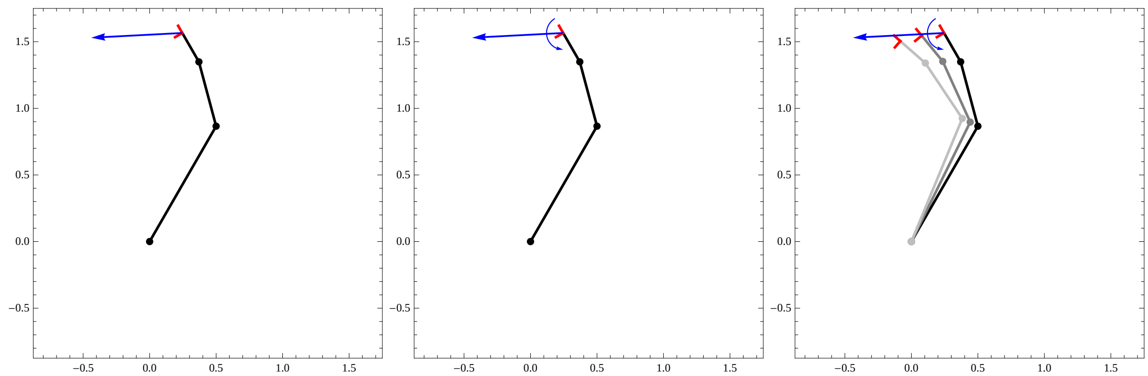
$$\left. \begin{array}{cc} -l_2 \sin(\alpha_1 + \alpha_2) - l_3 \sin(\alpha_1 + \alpha_2 + \alpha_3) & -l_3 \sin(\alpha_1 + \alpha_2 + \alpha_3) \\ l_2 \cos(\alpha_1 + \alpha_2) + l_3 \cos(\alpha_1 + \alpha_2 + \alpha_3) & +l_3 \cos(\alpha_1 + \alpha_2 + \alpha_3) \\ \hline 1 & 1 \end{array} \right].$$

(b) This part simply asks that we evaluate the Jacobian and multiply it by the joint velocities as per

$$v = J(\alpha)\dot{\alpha} = J(\left\{ \begin{array}{c} 1.05 \\ 0.785 \\ 0.262 \end{array} \right\}) \cdot \left\{ \begin{array}{c} 0.262 \\ 0.393 \\ 0 \end{array} \right\} = \left[\begin{array}{ccc} -1.565 & -0.699 & -0.217 \\ 0.246 & -0.254 & -0.125 \\ \hline 1.000 & 1.000 & 1.000 \end{array} \right] \left\{ \begin{array}{c} 0.262 \\ 0.393 \\ 0. \end{array} \right\} = \left\{ \begin{array}{c} -0.685 \\ -0.036 \\ 0.654 \end{array} \right\}$$

It will rotate counter-clockwise (positive angular rate).

(c) Figure 1 has a few depictions related to this problem. The left plot depicts the vector associated with the expected translational motion that would occur if the joint angles move from their current position with the velocity specified in (b). The middle plot depicts the same, but also shows the expected angular motion of the coordinate frame (counter-clockwise / positive movement). The right-most plot shows the expected end effector movement vector and also two additional manipulator configurations. These are for the two joint configurations, $\alpha + \frac{1}{8}\dot{\alpha}$ and $\alpha + \frac{1}{4}\dot{\alpha}$. Note that the movement of the manipulator end effector is roughly predicted by the vector. As the joint configuration update gets bigger, the movement starts to stray from the prediction. The prediction is a linear approximation that is only valid for small motions.



(a) Position Manipulator Jacobian (b) Manipulator Jacobian (c) α update versus Jacobian vector

Figure 1: Plots for Problem 1c.

Problem 2. (60 pts) Let's consider again the three-link rotational planar manipulator from the previous homeworks, c.f. Figure 2. The link lengths were specified to be $l_1 = 1$, $l_2 = \frac{1}{2}$, and $l_3 = \frac{1}{4}$. We will investigate further what it means to generate trajectories using different concepts from the class (configuration-space, vector form versus Lie group form).

(a) In one of the last homeworks, you were asked to generte a cubic spline trajectory connecting a pair of inital and final joint angles. $\alpha(0) = (-\pi/6, \pi/4, -\pi/3)^T$ to the final joint angles $\alpha(5) = (0, -\pi/12, \pi/4)^T$. Compute the forward kinematics of these two joint angles to determine the associated end-effector initial and final configurations.

(b) Using the vector form for the configuration, $g = (x, y, \theta)^T$, connect the initial and final configurations with a straight line trajectory. Basically, the straight-line trajectory ignores the Lie group structure and treats the coordinates like independent vector elements. This is done by definiing the following trajectory

$$\left\{ \begin{array}{c} x_e^*(t) \\ y_e^*(t) \\ \theta_e^*(t) \end{array} \right\} = \left\{ \begin{array}{c} x_e^*(0) \\ y_e^*(0) \\ \theta_e^*(0) \end{array} \right\} + \frac{t}{T} \left(\left\{ \begin{array}{c} x_e^*(T) \\ y_e^*(T) \\ \theta_e^*(T) \end{array} \right\} - \left\{ \begin{array}{c} x_e^*(0) \\ y_e^*(0) \\ \theta_e^*(0) \end{array} \right\} \right).$$

For now, break up the trajectory into two segments with the same time duration (one way-point). You will need to apply the inverse kinematics to the trajectory end-points of the segments to get the desired joint-angles (be careful with multiple solutions here!), then spline the desired joint-angles together. What are the spline functions for each joint angle?

(c) Instead of vectorizing the group space and treating it like a linear space, lets try to design a trajectory that is more suited to the Lie group. Using the twist $\xi \in \mathfrak{se}(2)$ associated to the transformation between the initial and final configurations, the trajectory we are going to try to follow is

$$g_e^*(t) = g_e^*(0) \exp(\xi t)$$

Break the trajectory up into two segments along the exponential. You'll need to compute the half-way point along the exponential trajectory, compute the inverse kinematics for the segment end-points to get the desired joint angles, then spline the desired joint-angles together. What are the spline functions for each joint angle?

(d) Plot the joint trajectories, $\alpha(t)$, for the solutions to (b) and (c). Compare them against the joint trajectory from the last homework.

(e) Plot the final end-effector group trajectories, $g_e(t)$, for the solutions to (b) and (c). Since the trajectories you design will be in joint angles, you will have to transform the trajectory using the forward kinematics to get the end-effector Lie group configuration for plotting. This can be done nicely in the appropriate for loop. For (b) and (c) you have the desired trajectories that you can compare against. How well do your trajectories fit the desired trajectories?

(f) Comment on the differences between the three solutions regarding joint-space, vectorized configuration space, and exponential configuration-space. Comment on any challenges associated to computing the trajectories and finding the actual joint angles.

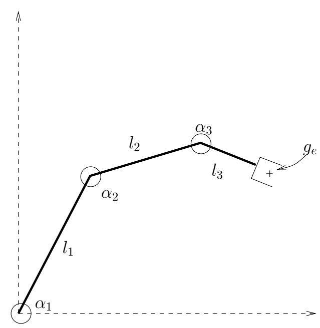


Figure 2: Planar 3R Manipulator.

Note: If you'd like to have more waypoints in the trajectory, then feel free to do so. You can also use Matlab's spline toolbox to handle the multiple waypoints, or you can solve using the matrix worked out in class.

Solution 2. This problem essentially asks for a cubic splined trajectory with one specific intermediate waypoint. According to the notes/lecture, one way to do that is to solve the matrix problem:

$$a = P_{simp}(T/2, T/2) \left\{ \begin{array}{c} p_i \\ p_v \\ p_f \end{array} \right\}$$

where the simplified polynomial solution matrix is

$$P_{simp}(t,t) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ -\frac{9}{4t^2} & \frac{3}{t^2} & -\frac{3}{4t^2} \\ \frac{5}{4t^3} & -\frac{2}{t^3} & \frac{3}{4t^3} \\ 0 & 1 & 0 \\ -\frac{3}{4t} & 0 & \frac{3}{4t} \\ \frac{3}{2t^2} & -\frac{3}{t^2} & \frac{3}{2t^2} \\ -\frac{3}{4t^3} & \frac{2}{t^3} & -\frac{5}{4t^3} \end{bmatrix}$$

All the is needed is to figure out what the initial p_i, intermediate p_v, and final p_f values should be. Note that there will be three such solutions, one for each joint angle, α_1, α_2, and α_3, and each requiring the initial, intermediate, and final values. The problem statement describes two different techniques, one related to the problems from the midterm and one from simply considering a vectorization of the end-effector configuration.

(a) The initial and final joint configurations are:

$$g_i = \begin{bmatrix} 0.707 & 0.707 & 1.53 \\ -0.707 & 0.707 & -0.547 \\ 0. & 0. & 1. \end{bmatrix} \quad \text{and} \quad g_f = \begin{bmatrix} 0.866 & -0.5 & 1.7 \\ 0.5 & 0.866 & -0.00441 \\ 0. & 0. & 1. \end{bmatrix}$$

(b) If the initial and final end-effector configurations are written in vector form instead of homogeneous form and treated like vectors, one candidate trajectory connecting the two end-effector configurations is

$$\left\{ \begin{array}{c} x_e^*(t) \\ y_e^*(t) \\ \theta_e^*(t) \end{array} \right\} = \left\{ \begin{array}{c} 1.52577 \\ -0.547367 \\ -0.785398 \end{array} \right\} + \frac{t}{T} \left\{ \begin{array}{c} 0.173704 \\ 0.542958 \\ 1.309 \end{array} \right\}.$$

The inverse kinematics were solved for in earlier homework, so we just have to loop through the end-effector waypoints and use that inverse kinematics solution (hopefully coded up as a Matlab function) to generate the joint angle waypoints. I picked the first solution for all of the end-effector waypoints. Including the initial and final end-effector configurations, the trajectory way points were

$$\left\{ \begin{array}{c} 1.53 \\ -0.547 \\ -0.785 \end{array} \right\}, \quad \left\{ \begin{array}{c} 1.61 \\ -0.276 \\ -0.131 \end{array} \right\}, \quad \left\{ \begin{array}{c} 1.7 \\ -0.00441 \\ 0.524 \end{array} \right\}$$

The associated joint angles are

$$\left\{ \begin{array}{c} -0.524 \\ 0.785 \\ -1.05 \end{array} \right\}, \quad \left\{ \begin{array}{c} -0.447 \\ 0.834 \\ -0.518 \end{array} \right\}, \quad \left\{ \begin{array}{c} 0. \\ -0.262 \\ 0.785 \end{array} \right\}$$

(d) The intermediate way-points are generated using the exponential trajectory,

$$g(t_k) = g_i \exp(\xi t_k), \quad \text{where } t_k \in (0, T).$$

The trajectory points, including initial and final end-effector configurations, are

$$\left\{ \begin{array}{c} 1.53 \\ -0.547 \\ -0.785 \end{array} \right\}, \quad \left\{ \begin{array}{c} 1.7 \\ -0.305 \\ -0.131 \end{array} \right\}, \quad \left\{ \begin{array}{c} 1.7 \\ -0.00441 \\ 0.524 \end{array} \right\}$$

and the associated joint angles are

$$\left\{ \begin{array}{c} -0.524 \\ 0.785 \\ -1.05 \end{array} \right\}, \quad \left\{ \begin{array}{c} -0.294 \\ 0.327 \\ -0.164 \end{array} \right\}, \quad \left\{ \begin{array}{c} 0. \\ -0.262 \\ 0.785 \end{array} \right\}.$$

(e) Plots of the joint configuration coordinate values along the trajectories are given in Figure 3.

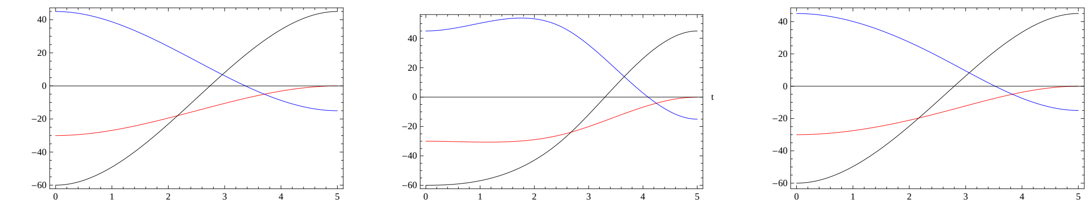
(f) Parametric plots of the trajectory with snapshots of the manipulator configuration are depicted in Figure 4.

(g) It is clear that the trajectories do not match each other, see Figure 6. This is to be expected. The linear trajectory that was followed in the joint configuration space results in a curved trajectory when mapped through the forward kinematics (a nonlinear function of the joint angles). Working with the vectorized version of $SE(2)$ does give a linear trajectory in the end-effector space, but notice that the way it has been splined. The spline cannot exactly fit the line due to the fact that the joint angles result in rotations, which gives curved trajectories.

The exponential is also different in a sense. It is not always the best choice, but it is a potential choice to use. Notice that it works not in the nonlinear group space $SE(2)$, but moves to the linear tangent space $\mathfrak{se}(2)$. A linear solution in $\mathfrak{se}(2)$ gives a nonlinear solution in $SE(2)$ (for this case at least), and also in the joint space. Again, the spline does not exactly fit, but is somewhat close for a portion of the trajectory. Increasing the number of spline points would reduce the trajectory matching error for both cases.

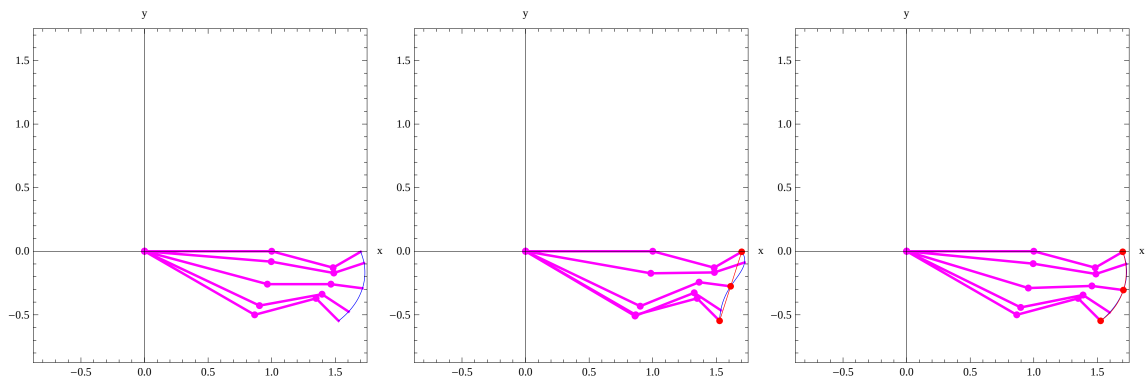
The exponential trajectory follows a circular path except when there is no rotation (then it is linear). One problem with this is that the trajectory arc can sometimes leave the manipulator workspace.

Problem 3. (20 pts) This is a reprisal of the earlier homework problem where you solved the solution to the forward kinematics for the Lynxmotion L6 manipulator using the product of Lie groups. This week the task is to compute the forward kinematics of the Lynxmotion L6 manipulator using the product of exponentials. Show that the same function results. Use the reference configuration to be the straight-up configuration depicted in Figure 7.



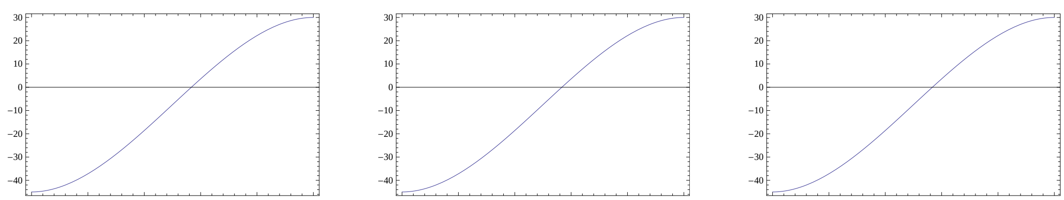
(a) Joint Spline from previous homework. (b) Linear Trajectory Splined. (c) Exponential Trajectory Splined.

Figure 3: Plot of $\alpha_1(t)$ [blue], $\alpha_2(t)$ [green], and $\alpha_3(t)$ [red].



(a) Joint Spline. (b) Linear Trajectory Spline. (c) Exponential Trajectory Spline.

Figure 4: Parametric plots of trajectory with snapshots of manipulator.



(a) Joint Spline. (b) Linear Trajectory Spline. (c) Exponential Trajectory Spline.

Figure 5: Orientation versus time plots of trajectory.

Solution 3. In order to apply the product of exponential approach for computing the forward kinematics, it is important to determine the Lie algebra elements (AKA the *twists*), ξ_i, associated with the joints. Prior to doing so, the base frame must be chosen and the reference configuration determined. The base frame is already depicted in Figure 7 as is the reference frame I'll be using for the purposes of this solution. That doesn't mean that you had to pick it (I personally use the straight up configuration for some of the manipulators). So, with that said, the base frame to reference frame end-effector configuration is:

$$g_o = \left[\begin{array}{c|c} \mathbb{1} & \left\{\begin{array}{c} 0 \\ l_1 + l_2 + l_3 + l_4 \\ l_0 \end{array}\right\} \\ \hline 0 & 1 \end{array}\right]$$

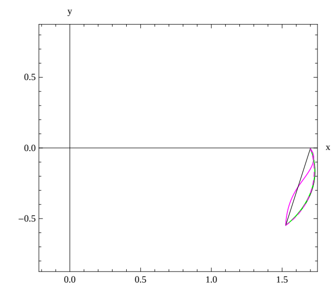


Figure 6: Parametric plots of all trajectories, actual vector (black), desired vector (magenta, solid), actual exponential (green), and desired exponential (magenta, dash-dotted).

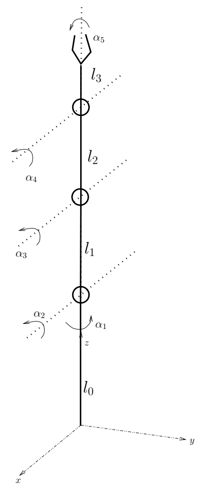


Figure 7: Lynxmotion L6 manipulator.

The corresponding twists are:

$$\xi_1 = \left\{ \frac{q_1 \times \omega_1}{\omega_1} \right\} = \left\{ \begin{array}{c} 0 \\ 0 \\ 0 \\ \hline 0 \\ 0 \\ 1 \end{array} \right\} \quad \text{where} \quad q_1 = \left\{ \begin{array}{c} 0 \\ 0 \\ 0 \end{array} \right\} \text{ and } \omega_1 = \left\{ \begin{array}{c} 0 \\ 0 \\ 1 \end{array} \right\}$$

$$\xi_2 = \left\{ \frac{q_2 \times \omega_2}{\omega_2} \right\} = \left\{ \begin{array}{c} 0 \\ l_0 \\ 0 \\ \hline 1 \\ 0 \\ 0 \end{array} \right\} \quad \text{where} \quad q_2 = \left\{ \begin{array}{c} 0 \\ l_0 \\ 0 \end{array} \right\} \text{ and } \omega_2 = \left\{ \begin{array}{c} 1 \\ 0 \\ 0 \end{array} \right\}$$

$$\xi_3 = \left\{\frac{q_3 \times \omega_3}{\omega_3}\right\} = \left\{\begin{array}{c} 0 \\ l_0 \\ -l_1 \\ \hline 1 \\ 0 \\ 0 \end{array}\right\} \quad \text{where} \quad q_3 = \left\{\begin{array}{c} 0 \\ l_1 \\ l_0 \end{array}\right\} \text{ and } \omega_3 = \left\{\begin{array}{c} 1 \\ 0 \\ 0 \end{array}\right\}$$

$$\xi_4 = \left\{\frac{q_4 \times \omega_4}{\omega_4}\right\} = \left\{\begin{array}{c} 0 \\ l_0 \\ -l_1 - l_2 \\ \hline 1 \\ 0 \\ 0 \end{array}\right\} \quad \text{where} \quad q_4 = \left\{\begin{array}{c} 0 \\ l_1 + l_2 \\ l_0 \end{array}\right\} \text{ and } \omega_4 = \left\{\begin{array}{c} 1 \\ 0 \\ 0 \end{array}\right\}$$

$$\xi_5 = \left\{\frac{q_5 \times \omega_5}{\omega_5}\right\} = \left\{\begin{array}{c} -l_0 \\ 0 \\ 0 \\ \hline 0 \\ 1 \\ 0 \end{array}\right\} \quad \text{where} \quad q_5 = \left\{\begin{array}{c} 0 \\ 0 \\ l_0 \end{array}\right\} \text{ and } \omega_5 = \left\{\begin{array}{c} 0 \\ 1 \\ 0 \end{array}\right\}$$

The exponential corresponding to a given twist, given rotation by α_i, is

$$e^{\hat{\xi}_i \alpha_i} = \left[\begin{array}{c|c} e^{\hat{\omega}_i \alpha_i} & \left(\left(\mathbb{1} - e^{\hat{\omega}_i \alpha_i}\right) \hat{\omega}_i + \alpha_i \omega_i \omega_i^T\right) v_i \\ \hline 0 & 1 \end{array}\right].$$

The corresponding exponential transformations are

$$e^{\hat{\omega}_1 \alpha_1} = \left[\begin{array}{ccc|c} \cos(\alpha_1) & -\sin(\alpha_1) & 0 & 0 \\ \sin(\alpha_1) & \cos(\alpha_1) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline \multicolumn{3}{c|}{0} & 1 \end{array}\right]$$

$$e^{\hat{\omega}_2 \alpha_2} = \left[\begin{array}{cccc|c} 1 & 0 & 0 & & 0 \\ 0 & \cos(\alpha_2) & -\sin(\alpha_2) & & l_0 \sin(\alpha_2) \\ 0 & \sin(\alpha_2) & \cos(\alpha_2) & 0 & l_0 - l_0 \cos(\alpha_2) \\ \hline \multicolumn{4}{c|}{0} & 1 \end{array}\right]$$

$$e^{\hat{\omega}_3 \alpha_3} = \left[\begin{array}{cccc|c} 1 & 0 & 0 & & 0 \\ 0 & \cos(\alpha_3) & -\sin(\alpha_3) & & l_1 + l_0 \sin(\alpha_3) - l_1 \cos(\alpha_3) \\ 0 & \sin(\alpha_3) & \cos(\alpha_3) & 0 & l_0 - l_0 \cos(\alpha_3) - l_1 \sin(\alpha_3) \\ \hline \multicolumn{4}{c|}{0} & 1 \end{array}\right]$$

$$e^{\hat{\omega}_4 \alpha_4} = \left[\begin{array}{cccc|c} 1 & 0 & 0 & & 0 \\ 0 & \cos(\alpha_4) & -\sin(\alpha_4) & & (l_1 + l_2) + l_0 \sin(\alpha_4) - (l_1 + l_2) \cos(\alpha_4) \\ 0 & \sin(\alpha_4) & \cos(\alpha_4) & 0 & l_0 - l_0 \cos(\alpha_4) - (l_1 + l_2) \sin(\alpha_4) \\ \hline \multicolumn{4}{c|}{0} & 1 \end{array}\right]$$

$$e^{\hat{\omega}_5 \alpha_5} = \left[\begin{array}{ccc|c} \cos(\alpha_1) & 0 & \sin(\alpha_1) & -l_0 \sin(\alpha_5) \\ 0 & 1 & 0 & 0 \\ -\sin(\alpha_1) & 0 & \cos(\alpha_1) & l_0 - l_0 \cos(\alpha_5) \\ \hline \multicolumn{3}{c|}{0} & 1 \end{array}\right]$$

Under the product of exponentials, the forward kinematics are

$$g_e(\alpha) = e^{\xi_1 \alpha_1} e^{\xi_2 \alpha_2} e^{\xi_3 \alpha_3} e^{\xi_4 \alpha_4} e^{\xi_5 \alpha_5} g_0.$$

The solution works out to be the same as the product of Lie groups, with the rotation matrix and displacement vector

$$R_e(\alpha) = \begin{bmatrix} \cos(\alpha_1)\cos(\alpha_5) + \sin(\alpha_1)\sin(\alpha_2+\alpha_3+\alpha_4)\sin(\alpha_5) & -\sin(\alpha_1)\cos(\alpha_2+\alpha_3+\alpha_4) & \dots \\ \sin(\alpha_1)\cos(\alpha_5) - \cos(\alpha_1)\sin(\alpha_2+\alpha_3+\alpha_4)\sin(\alpha_5) & \cos(\alpha_1)\cos(\alpha_2+\alpha_3+\alpha_4) & \dots \\ \cos(\alpha_2+\alpha_3+\alpha_4)\sin(\alpha_5) & \sin(\alpha_2+\alpha_3+\alpha_4) & \end{bmatrix}$$

$$\begin{matrix} \cos(\alpha_1)\sin(\alpha_5) + \sin(\alpha_1)\sin(\alpha_2+\alpha_3+\alpha_4)\cos(\alpha_5) \\ \cos(\alpha_1)\sin(\alpha_5) + \sin(\alpha_1)\sin(\alpha_2+\alpha_3+\alpha_4)\cos(\alpha_5) \\ \cos(\alpha_2+\alpha_3+\alpha_4)\cos(\alpha_5) \end{matrix}\Bigg]$$

and

$$d_e(\alpha) = \left\{ \begin{matrix} -\sin(\alpha_1)\big(l_1\cos(\alpha_2) + l_2\cos(\alpha_2+\alpha_3) + (l_3+l_4)\cos(\alpha_2+\alpha_3+\alpha_4)\big) \\ \cos(\alpha_1)\big(l_1\cos(\alpha_2) + l_2\cos(\alpha_2+\alpha_3) + (l_3+l_4)\cos(\alpha_2+\alpha_3+\alpha_4)\big) \\ l_0 + l_1\sin(\alpha_2) + l_2\sin(\alpha_2+\alpha_3) + (l_3+l_4)\sin(\alpha_2+\alpha_3+\alpha_4) \end{matrix} \right\},$$

respectively.

Lab Assignments: For the traditional manipulator problems, there are actually two things to accomplish. One is to continue with the `piktul` manipulator and the other is to get started on the `lynx6` manipulator.

Problem 4: Manipulator. (20 pts) Work out the next piktul adventure (Set 1, Module 5), which continues exploring resolved rate trajectory generation. Reading it, you should note that we will be working with the full end-effector configuration in vector form. This is safe since the orientation is only about one axis.

The trajectory should start at the initial joint configuration of:

$$\alpha_i = (1.75, -30, 18, -25, 0.45)^T$$

and this time move at the linear and angular velocity

$$v_{des} = \left\{ \begin{matrix} -0.4 \\ 2.5 \\ -0.25 \end{matrix} \right\} \text{inches/sec} \quad \text{and} \quad \omega_{des} = 15^\circ/\text{sec}$$

for 3 seconds. Alternatively, we can think of this as moving at the velocity:

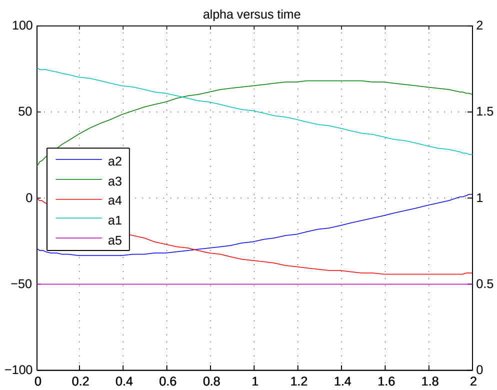
$$\dot{\vec{g}} = (-0.4, 2.5, -0.25, 15)^T$$

where the first three coordinates are the linear velocity and the last coordinate is the angular velocity.

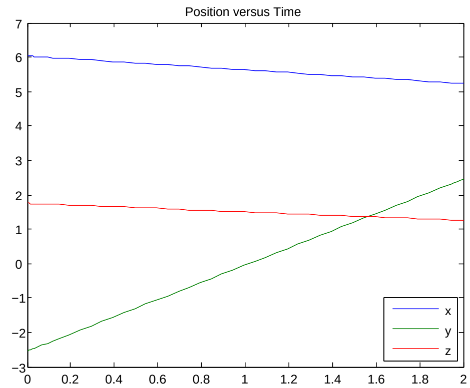
Solution 4. The solution to this involves extending the position based one to include the rotation. copying the position based Jacobian and renaming it to be

First, the code requires a Jacobian computation member function, which I've called `Jacobian` and computes the Jacobian for the vector form as specified:

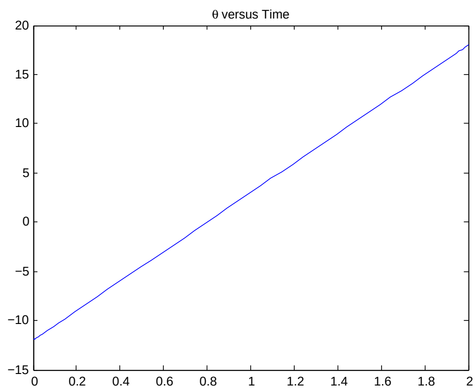
```
function mJ = Jacobian(this, alpha)

s1 = sind(alpha(2));
c1 = cosd(alpha(2));

s12 = sind(alpha(2)+alpha(3));
c12 = cosd(alpha(2)+alpha(3));

mJ = [ 0, -this.linklen(1)*s1 - this.linklen(2)*s12 , ...
                                        -this.linklen(2)*s12 , 0 , 0 ; ...
       0,  this.linklen(1)*c1 + this.linklen(2)*c12 , ...
                                         this.linklen(2)*c12 , 0 , 0 ; ...
```



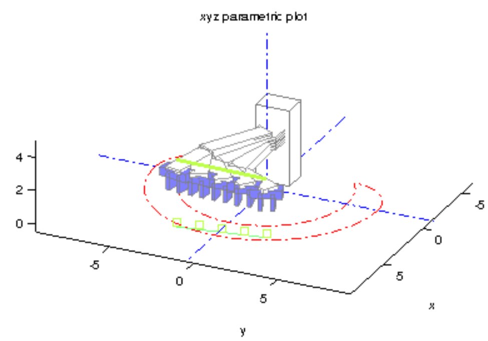
(a) α versus t.



(b) (x, y, z) versus t.



(c) θ versus t.



(d) Parametric plot w/manipulator snapshots.

Figure 8: Plots of the joints versus time, the position versus time (to show that it is indeed a line) using the forward kinematics, the orientation versus time, and a visualization of the trajectory. If your code was working, then it should have followed the trajectory depicted in (d).

```
           1,          0          ,       0    , 0 , 0 ; ...
           0,          1          ,       1    , 1 , 0 ];

  end
```

The rest pretty much stays the same. I did copy the `genPositionTrajectory` function and rename it to `genTrajectory` (plus I renamed the ODE function to `vecODE`) since the arguments are different (in terms of their meaning and dimension). Also, to map the joint angles into end-effector configurations for plotting, the forward kinematics function was augmented to include the orientation of the end-effector (just the sum of α_2, α_3, and α_4). The `followJointTrajectory` function was recycled from last week; no changes were necessary.

Plots of the results can be found in Figure 8. The main script that did all of the variable setup and function calling was pretty much the same, except that it called `genTrajectory` with the desired trajectory definitions, which looked like:

```
ptraj.position = @(t)[];                        % Not really used for this prob.
ptraj.velocity = @(t)[-0.4; 2.5; -0.25; 15]; % Constant velocity.
ptraj.tspan = [0, 10];                          % Arbitrary, but big enough tspan.
```

Notice that the angular rate is given in degrees per second. Conversion to radians is needed in the `vecODE` function in order to track the orientation properly.

Problem 4: Manipulator. (20 pts) Given the dimensions of the L6 manipultor you computed from prior homeworks and the forward kinematics with the home configuration that you chose, you are to write a program to pick and place a object. To do this you will need to complete the inverskin function at the end of the lynx6 m-file. Make sure that both the forward kinematics and inverse kinematics functions conform to your particular manipulator.

You must turn in the code for the inverse kinematics and the matlab script file for accomplishing the above. Normally, I would give you the code stubs for this set of actions, but I have already given you something similar in previous homeworks. When combined with the recipe given in the Lynx-6 Adventures wiki above plus a few quick changes, you should be good to go.

The initial and final configurations are:

$$g_i = \left[\begin{array}{ccc|c} 0.7071 & -0.7071 & 0 & 5 \\ -0.7071 & -0.7071 & 0 & 5 \\ 0 & 0 & -1 & 0.35 \\ \hline \multicolumn{3}{c|}{0} & 1 \end{array}\right]. \quad \text{and} \quad g_f = \left[\begin{array}{ccc|c} 1 & 0 & 0 & -4.4 \\ 0 & -1 & 0 & 5.5 \\ 0 & 0 & -1 & 0.25 \\ \hline \multicolumn{3}{c|}{0} & 1 \end{array}\right]$$

Good luck.

Solution 4. Before working out the inverse kinematics, we need to first define the appropriate forward kinematics relating to the home position defined, e.g., straight up. Using the product of Lie groups,

$$g_e(\alpha) = g_1(\alpha_1)g_2(\alpha_2)g_3(\alpha_3)g_4(\alpha_4)g_5(\alpha_5)g_6,$$

where each transform g_i goes from joint $(i-1)$ to joint i. The different g_i's are:

$$g_1(\alpha_1) = \left[\begin{array}{ccc|c} \cos(\alpha_1) & -\sin(\alpha_1) & 0 & 0 \\ \sin(\alpha_1) & \cos(\alpha_1) & 0 & 0 \\ 0 & 0 & 1 & l_0 \\ \hline \multicolumn{3}{c|}{0} & 1 \end{array}\right], \qquad g_2(\alpha_2) = \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & \cos(\alpha_2) & -\sin(\alpha_2) & 0 \\ 0 & \sin(\alpha_2) & \cos(\alpha_2) & 0 \\ \hline \multicolumn{3}{c|}{0} & 1 \end{array}\right],$$

$$g_3(\alpha_3) = \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & \cos(\alpha_3) & -\sin(\alpha_3) & 0 \\ 0 & \sin(\alpha_3) & \cos(\alpha_3) & l_1 \\ \hline \multicolumn{3}{c|}{0} & 1 \end{array}\right], \qquad g_4(\alpha_4) = \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & \cos(\alpha_4) & -\sin(\alpha_4) & 0 \\ 0 & \sin(\alpha_4) & \cos(\alpha_4) & l_2 \\ \hline \multicolumn{3}{c|}{0} & 1 \end{array}\right],$$

$$g_5(\alpha_5) = \left[\begin{array}{ccc|c} \cos(\alpha_5) & -\sin(\alpha_5) & 0 & 0 \\ \sin(\alpha_5) & \cos(\alpha_5) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline \multicolumn{3}{c|}{0} & 1 \end{array}\right], \qquad g_6 = \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & l_3 \\ \hline \multicolumn{3}{c|}{0} & 1 \end{array}\right],$$

OK, with this we can begin to work out the inverse kinematics solution in a manner inspired by Pieper's solution.

(a) The first step is to separate the "arm" part from the "hand" part. Thus, consider $g_e(\alpha)$ to be composed of a transformation that goes from the base to the wrist, $g_w(\alpha)$, and a transformation that goes from the wrist to the hand $g_h(\alpha)$, whereby $g_e(\alpha) = g_w(\alpha)g_h(\alpha)$. To do this, we are going to have to break up the Lie group products into two parts. Of note, the transformation g_4 must be broken up into $g_4(\alpha) = \bar{g}_4\tilde{g}_4(\alpha)$, where $\bar{g}_4$ is the constant part and $\tilde{g}_4(\alpha)$ is the part that changes with the joint configuration. Under the desired decomposition,

$$g_w(\alpha) = g_1(\alpha_1)g_2(\alpha_2)g_3(\alpha_3)\bar{g}_4, \qquad \text{and} \qquad g_h(\alpha) = \tilde{g}_4(\alpha_4)g_5(\alpha_5)g_6.$$

We need to find the joint configuration α leading to the desired end-effector configuration g_e^*,

$$g_e^* = g_w(\alpha)g_h(\alpha).$$

Now, recall that the last transformation of $g_h(\alpha)$ is g_6 which is constant, and the remaining transformations g_4 and g_5 do not have a translational component. A logical first step would be to (right) invert both sides of the equation by this constant transformation left hand side leading the desired wrist position,

$$g_w^* \equiv g_e^* g_6^{-1}.$$

The wrist positioning objective is to find α_1^*, α_2^*, and α_3^* such that

$$Position(g_w^*) = Position(g_w(\alpha^*)).$$

The solution for the third joint angle is

$$\alpha_3 = \pm \cos^{-1}\left(\frac{x_w^2 + y_w^2 + (z_w - l_0)^2 - l_1^2 - l_2^2}{2l_1 l_2}\right).$$

Note that due to the plus or minus answer, there are now two possible solutions.

For the second joint angle, we know that the square of the first two components of the wrist position is equal to

$$(x_w(\alpha))^2 + (y_w(\alpha))^2 = (l_1 \sin(\alpha_2) + l_2 \sin(\alpha_2 + \alpha_3))^2 .$$

If we define $r \equiv \sqrt{(x_w^*)^2 + (y_w^*)^2}$, then we seek the joint angle α_2^* such that

$$r = \pm\left(l_1 \sin(\alpha_2) + l_2 \sin(\alpha_2 + \alpha_3)\right),$$

where the plus or minus is due to the sign ambiguity of the square root (there are now four possible solutions to choose from). This is because sine is an odd function ($\sin(\pm\alpha) = \pm\sin(\alpha)$, which ultimately affects the arctangent function. For the second joint angle, let $w = \tan(\alpha_2/2)$. Under this substituion, we have

$$\cos(\alpha_2) = \frac{1-w^2}{1+w^2} \quad \text{and} \quad \sin(\alpha_2) = \frac{2w}{1+w^2}.$$

Applying the substition, we now seek the solution to

$$(r \pm l_2 \sin(\alpha_3))w^2 \mp 2(l_1 + l_2 \cos(\alpha_3))w + (r \mp l_2 \sin(\alpha_3)) = 0.$$

Once the solution to the quadratic equation is found, the correct solution of the two quadratic solutions must be chosen and inverted to get, $\alpha_2 = 2\tan^{-1}(w)$. Note that there are now four potential solutions, not all of which are correct.

Finally, for the first joint angle, we have

$$\begin{aligned} x_w(\alpha) &= (l_1 \sin(\alpha_2) + l_2 \sin(\alpha_2 + \alpha_3)) \sin(\alpha_1) \\ y_w(\alpha) &= -\left(l_1 \sin(\alpha_2) + l_2 \sin(\alpha_2 + \alpha_3)\right) \cos(\alpha_1). \end{aligned}$$

Therefore, given the desired wrist position, we solve for α_1^* such that

$$\begin{aligned} x_w^* &= (l_1 \sin(\alpha_2) + l_2 \sin(\alpha_2 + \alpha_3)) \sin(\alpha_1) \\ y_w^* &= -\left(l_1 \sin(\alpha_2) + l_2 \sin(\alpha_2 + \alpha_3)\right) \cos(\alpha_1). \end{aligned}$$

Plugging the known values for α_2 and α_3, we get

$$\begin{aligned} \frac{x_w^*}{l_1 \sin(\alpha_2) + l_2 \sin(\alpha_2 + \alpha_3)} &= \sin(\alpha_1) \\ -\frac{y_w^*}{l_1 \sin(\alpha_2) + l_2 \sin(\alpha_2 + \alpha_3)} &= \cos(\alpha_1) \end{aligned}$$

The solution involves the arctangent function,

$$\alpha_1 = \arctan\left(-\frac{y_w^*}{l_1 \sin(\alpha_2) + l_2 \sin(\alpha_2 + \alpha_3)}, \frac{x_w^*}{l_1 \sin(\alpha_2) + l_2 \sin(\alpha_2 + \alpha_3)}\right).$$

Once we have $g_w(\alpha)$, then we need to find the orientation part. Following the notes, we need to find the remaining α_4^* and α_5^* such that

$$g_e^* = g_w(\alpha^*) g_h(\alpha^*).$$

Since α_1^*, α_2^*, and α_3^* are known,

$$g_w^{-1}(\alpha^*)g_e^* = g_h(\alpha^*) = \tilde{g}_4(\alpha_4^*)g_5(\alpha_5^*)g_6.$$

Isolating the unknown components,

$$g_w^{-1}(\alpha^*)g_e^* g_6^{-1} = \tilde{g}_4(\alpha_4^*)g_5(\alpha_5^*).$$

At this point, both sides of the above equation should have zero translation. All that is left is to solve for the orientation. The rotation matrix R_h^* is something that will be computed based on given values. Therefore, we only need worry about the rotational portions of the above equation, which will involve a R_h^* from the left hand side and $R_h(\alpha) = R_4(\alpha_4)R_5(\alpha_5)$ from the right hand side, for which we have,

$$\begin{aligned}
R_h(\alpha) &= R_4(\alpha_4)R_5(\alpha_5) \\
&= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha_4) & -\sin(\alpha_4) \\ 0 & \sin(\alpha_4) & \cos(\alpha_4) \end{bmatrix} \cdot \begin{bmatrix} \cos(\alpha_5) & -\sin(\alpha_5) & 0 \\ \sin(\alpha_5) & \cos(\alpha_5) & 0 \\ 0 & 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} \cos(\alpha_5) & -\sin(\alpha_5) & 0 \\ \cos(\alpha_4)\sin(\alpha_5) & \cos(\alpha_4)\cos(\alpha_5) & -\sin(\alpha_4) \\ \sin(\alpha_4)\sin(\alpha_5) & \sin(\alpha_4)\cos(\alpha_5) & \cos(\alpha_4) \end{bmatrix}.
\end{aligned}$$

Note that form of this matrix is slightly different from that of the elbow manipulator, in the sense that there <u>should</u> be zero elements in the matrix. The symbolic form of the desired hand rotation matrix should be of the form

$$R_h^* = \begin{bmatrix} R_{11} & R_{12} & 0 \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix}$$

Further, one could also argue that $R_{11}^2 + R_{12}^2 = 1$ and $R_{23}^2 + R_{33}^2 = 1$ must both hold, as well as that the determinant of the lower-left 2×2 sub-matrix should be zero. Anyhow, comparing the above two matrices, one sees that it is possible to solve for the two final joints angles without any ambiguity. They are,

$$\begin{aligned}
\alpha_5^* &= \text{atan2}(-R_{12}, R_{11}), \text{ and} \\
\alpha_4^* &= \text{atan2}(-R_{23}, R_{33}),
\end{aligned}$$

where the atan2 function is as per Matlab.

(b) Using the above solution for the inverse kinematics, we get (in radians)

$$\alpha_i = (\frac{\pi}{6}, -\frac{\pi}{2}, \frac{\pi}{6}, -\frac{2\pi}{3}, 0)^T, \quad (\frac{\pi}{6}, -\frac{\pi}{3}, -\frac{\pi}{6}, -\frac{\pi}{2}, 0)^T, (-\frac{5\pi}{6}, \frac{\pi}{3}, \frac{\pi}{6}, \frac{\pi}{2}, \pi)^T, \quad \text{or} \quad (-\frac{5\pi}{6}, \frac{\pi}{2}, -\frac{\pi}{6}, \frac{2\pi}{3}, \pi)^T,$$

and

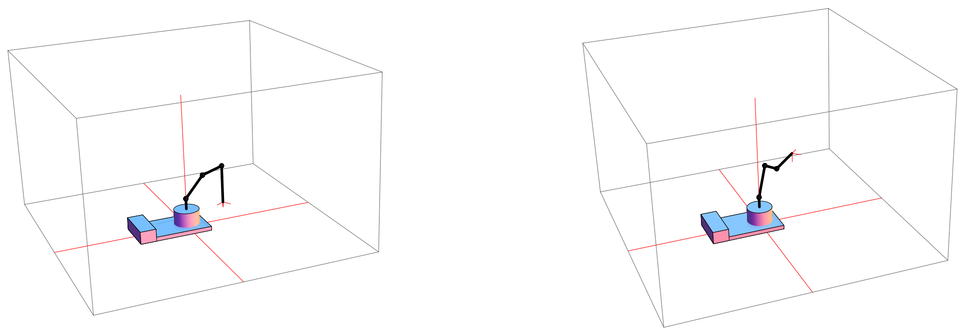
$$\alpha_f = (\frac{\pi}{4}, -\frac{\pi}{6}, -\frac{\pi}{2}, \frac{\pi}{6}, 0)^T, (\frac{\pi}{4}, -\frac{2\pi}{3}, \frac{\pi}{2}, -\frac{\pi}{3}, 0)^T, (-\frac{3\pi}{4}, \frac{4\pi}{3}, -\frac{\pi}{2}, \frac{\pi}{3}, \pi)^T, \quad \text{or} \quad (-\frac{3\pi}{4}, \frac{\pi}{6}, \frac{\pi}{2}, -\frac{\pi}{6}, \pi)^T,$$

respectively for the straight-up home configuration. In degrees, the solution would be

$$\begin{aligned}
\alpha_i = (30, -90, 30, -120, 0)^T, (30, -60, -30, 90, 0)^T, \\
(-150, 60, 30, 90, 180)^T, \text{ or } (-150, 90, -30, 120, 180)^T,
\end{aligned}$$

and

$$\begin{aligned}
\alpha_f = (45, -30, -90, 30, 0)^T, (45, -120, 90, -60, 0)^T, \\
(-135, 240, -90, 60, 180)^T, \text{ or } (-135, 30, 90, -30, 180)^T,
\end{aligned}$$



(a) Solution 2 to g_i. (b) Solution 1 to g_f.

Figure 9: Particular solutions to inverse kinematics of g_i and g_f. .

Note that some of these angles are not feasible for the lynxmotion and its joint angle limits. Multiple solutions help one still get to the answer in spite of phyical constraints. It's a curse for you in the homework, but a blessing when you really have to use it for something practical. Visualizations of the manipulators for the two configurations are given in Figure 9.

Note: Your answers may vary by a little bit from these. I used my own personal link lengths of

$$l_0 = 4.35433, l_1 = 4.72441, l_2 = 4.72441, \text{ and } l_3 = 5.11811 + 0.787402.$$

If you used the link lengths specified for the manipulator Jacobian problem, then the solutions would have looked like:

$$\alpha_i = (30, -98.4, 37.2, -119, 0)^T, \quad (30, -61.2, -37.2, -81.6, 0)^T, \\ (-150, 98.4, -37.2, 119, -180)^T, \quad \text{or} \quad (-150, 61.2, 37.2, 81.6, -180)^T,$$

and

$$\alpha_f = (45, -115, 76.1, -51.5, 0)^T, \quad (45, -38.5, -76.1, 24.6, 0)^T, \\ (-135, 115, -76.1, 51.5, 180)^T, \quad \text{or} \quad (-135, 38.5, 76.1, -24.6, -180)^T.$$

Problem 4: Turtlebot. (40 pts) Moving to the next module in the Turtlebot adventures, *Sensing*, work out the first three items in the list. These basically start to understand what visual sensor information is available, display it, and perform basic processing of the visual data. The processing emphasis is on the color data.

Solution 4. The hints should have hopefully given you a fair idea as to how to start. After that, it is a matter of figuring out how to identify the proper color to select, and the method. The easiest is to use the OpenCV `inRange` function. Doing a google search for the terms "OpenCV python color detection" should give a fair amount of hits on how to perform that, maybe even some tips for selecting the range so that you can be robust to lighting variation.

It is common for students to identify the color coordinates, then select a really tight range, to the point of being useless (only one very particular place in the room and orientation of the object will work). That's not the point.