ME 115(b): Homework #3
(Due January 30, 2000)

Problem 1: This problem is almost entirely computational. You can program in your favorite language. Most people will wishto use Mathematica.

Consider the three link planar robot shown in Figure 1

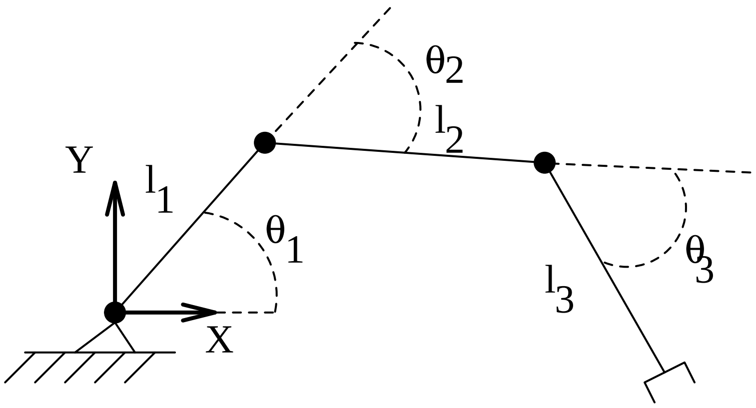

Figure 1: 3R planar "redundant" robot

This robot has three revolute joints. If we are only concerned with positioning the end-effector in the plane, and not concerned about the orientation of the last link, then this robot is redundant. It is the simplest redundant robot known, and often used as a toy example. Assume that $l_1 = l_2 = l_3 = 1$.

(a) Compute the forward kinematics, $f(\vec{\theta})$ of this manipulator (in symbolic form). Here, we are only concerned with the position of the tool frame origin, and not the tool frame's orientation.

(b) Compute the "hybrid" Jacobian matrix of this manipulator (in symbolic form). That is, compute $\partial f(\vec{\theta})/\partial \vec{theta}$.

(c) Let the desired end-effector trajectory be a circle centered at $(x, y) = (1.5, 0.0)$, with a radius of 0.5. One such way to parametrize this circle is:

$$\begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = \begin{bmatrix} 1.5 - 0.5\cos(\omega t) \\ 0.0 - 0.5\sin(\omega t) \end{bmatrix}$$

where ω controls the speed of the trajectory. Pick ω to be a reasonble number, such as π, or $\pi/2$.

Simulate this robot following the circular trajectory using the resolved rate trajectory planning scheme with a simple pseudo-inverse solution:

$$\dot{\theta} = J^{\#}(\vec{\theta})\dot{\vec{x}}(t)$$

where $J^{\#}$ is the pseudo-inverse of the hybrid Jacobian J, and where $\dot{\vec{x}}(t)$ is derived from the circular trajectory. You can use simple Euler integration to integrate the pseudo-inverse, or savvy mathematica hackers can use NDSolve to integrate the equations.

The output of your simulation should be:

(1) A plot of the joint angles versus time.

(2) A plot of the actual end-effector position versus time.

(d) (extra credit) Increase the radius of the above trajectory circle to 1.49 (so that it nearly touches the workspace boundary). Repeat the above simulation. Now repeat the above simulation using a damped pseudo-inverse solution.