SYSM 6305 Optimization Theory and Practice
Fall 2019

Project # 1   Due Date November 14, 2019

In this project you will investigate and compare the Newton-Raphson and Gradient Descent methods for the problem of numerical inverse kinematics on a two-link robot arm. Figure 1 shows a so-called SCARA robot arm, which is used in many assembly operations. (The term SCARA stands for Selective Compliant Articulated Robot Arm.) Figure 2 shows

Figure 1: SCARA Robot Arm

a schematic of the first two links of this robot. The coordinates (x_1, x_2) of the end of the

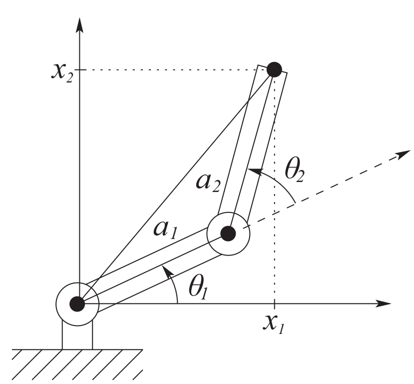

Figure 2: Two-link robot

second link, called **end effector** coordinates, are given in terms of the joint angles (θ_1, θ_2) by

$$x_1 = a_1 \cos\theta_1 + a_2 \cos(\theta_1 + \theta_2) \quad (1)$$
$$x_2 = a_1 \sin\theta_1 + a_2 \sin(\theta_1 + \theta_2) \quad (2)$$

where a_1 and a_2 are the lengths of the respective links.

The function $f(\theta_1, \theta_2) : \Re^2 \to \Re^2$ defined by Equations (1) and (2), is called the **forward kinematics**. The Jacobian of the forward kinematics function is given by

$$J(\theta_1, \theta_2) = \begin{bmatrix} -a_1 \sin\theta_1 - a_2 \sin(\theta_1 + \theta_2) & -a_2 \sin(\theta_1 + \theta_2) \\ a_1 \cos\theta_1 + a_2 \cos(\theta_1 + \theta_2) & a_2 \cos(\theta_1 + \theta_2) \end{bmatrix}$$

In order to plan the motion of a robot to complete a task we need the **inverse kinematics**, which is to determine the joint variables θ_1, θ_2 in terms of given values of x_1 and x_2. Since the forward kinematic equations are nonlinear, a solution may not be easy to find, nor is there a unique solution in general. We can see in the case of a two-link planar mechanism that there may be no solution, for example if the given (x_1, x_2) coordinates are out of reach of the manipulator.

If the given (x_1, x_2) coordinates are within the manipulator's reach there generally two solutions as shown in Figure 3, the so-called **left arm** and **right arm** configurations, or

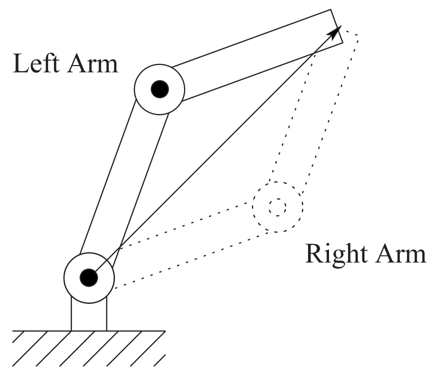

Figure 3: The two-link SCARA robot has two solutions to the inverse kinematics except at singular configurations, the left arm solution and the right arm solution.

there may be exactly one solution if the manipulator must be fully extended to reach the point. There may even be an infinite number of solutions in some cases. Can you say when?

Using the **law of cosines** we see from Figure 2 that the angle θ_2 is given by

$$\cos\theta_2 = \frac{x_1^2 + x_2^2 - a_1^2 - a_2^2}{2a_1a_2} := D \tag{3}$$

Since $\sin(\theta_2)$ is given as

$$\sin(\theta_2) = \pm\sqrt{1 - \cos^2(\theta_2)} = \pm\sqrt{1 - D^2} \tag{4}$$

we can compute θ_2 as

$$\theta_2 = \tan^{-1} \frac{\pm\sqrt{1 - D^2}}{D} \tag{5}$$

Both the left arm and right arm solutions are recovered by choosing the negative and positive signs in Equation (5), respectively. It is left as an exercise to show that θ_1 is now given as

$$\theta_1 = \tan^{-1}(x_2/x_1) - \tan^{-1}\left(\frac{a_2 \sin\theta_2}{a_1 + a_2 \cos\theta_2}\right) \tag{6}$$

Notice that the angle θ_1 depends on θ_2. This makes sense physically since we would expect to require a different value for θ_1, depending on which solution is chosen for θ_2. For a general robot with multiple links, the inverse kinematics problem can be quite difficult and so numerical search techniques are frequently employed.

Finding Inverse Kinematics Solutions

Let $x^d = (x_1^d, x_2^d) \in \Re^2$ be a vector specifying the desired end-effector coordinates and let $q = (q_1, q_2) = (\theta_1, \theta_2)$ be the vector of joint coordinates. If we set

$$G(q) = f(q) - x^d \tag{7}$$

where $f(q)$ is the forward kinematics function, then a solution to the inverse kinematics is a configuration q^d satisfying

$$G(q^d) = f(q^d) - x^d = 0.$$

A numerical solution to find a solution to the inverse kinematics problem is thus an algorithm of the form

$$q_{k+1} = q_k - \alpha_k M(q_k)(f(q_k) - x^d) \tag{8}$$

where M is a suitable 2×2 matrix and α_k is the step size.

Assignment

1. Show that the choice $M(q_k) = J^{-1}(q_k)$ in Equation (8) results from applying the Newton-Raphson method to find a root of $G(q)$ in Equation 7.

2. Write a Matlab program to find the inverse kinematics using the Newton-Raphson method. Assume that the link lengths a_1 and a_2 are equal to one.

3. Test your program for desired end-effector coordinates $x^d = (0.1, 1.5)$ with initial guess $q_1 = 0.2$, $q_2 = 0.6$. Experiment with different choices of step size.

4. How many iterations are needed to converge to within four decimal places of the exact solution given by Equations (5) and (6)?

5. Now consider the quadratic program

$$\min_q \; (f(q) - x^d)^T (f(q) - x^d)$$

   Show that a Gradient Descent method for this problem leads to Equation (8) with $M(q_k) = J^T(q_k)$.

6. Write a Matlab program to find the inverse kinematics using the Gradient Descent Method

7. Repeat the above steps 3 and 4 using the Gradient Descent method.

8. You may include the animation code provided on the course webpage to visualize your solutions.

9. Submit your Matlab code and your output. Clearly document your code.