Lab 1: FK and IK of a Kinematic Linkage

Joel Park

EE 183DA: Design of Robotic Systems I

UID: 204497509

January 26, 2017

Introduction:

A kinematic linkage is a robot that has a set of rigid bodies called links interconnected by a set of joints to a fixed base and a special part that performs specific tasks called the end-effector. Implementations of these kinematic linkages in the real world are remarkably mundane, from uses in industries to make manufacturing of products more facilitative for users to uses in hospitals to perform difficult surgical operations that require a more meticulous approach. There is no doubt that these robots are extremely useful in a variety of applications.

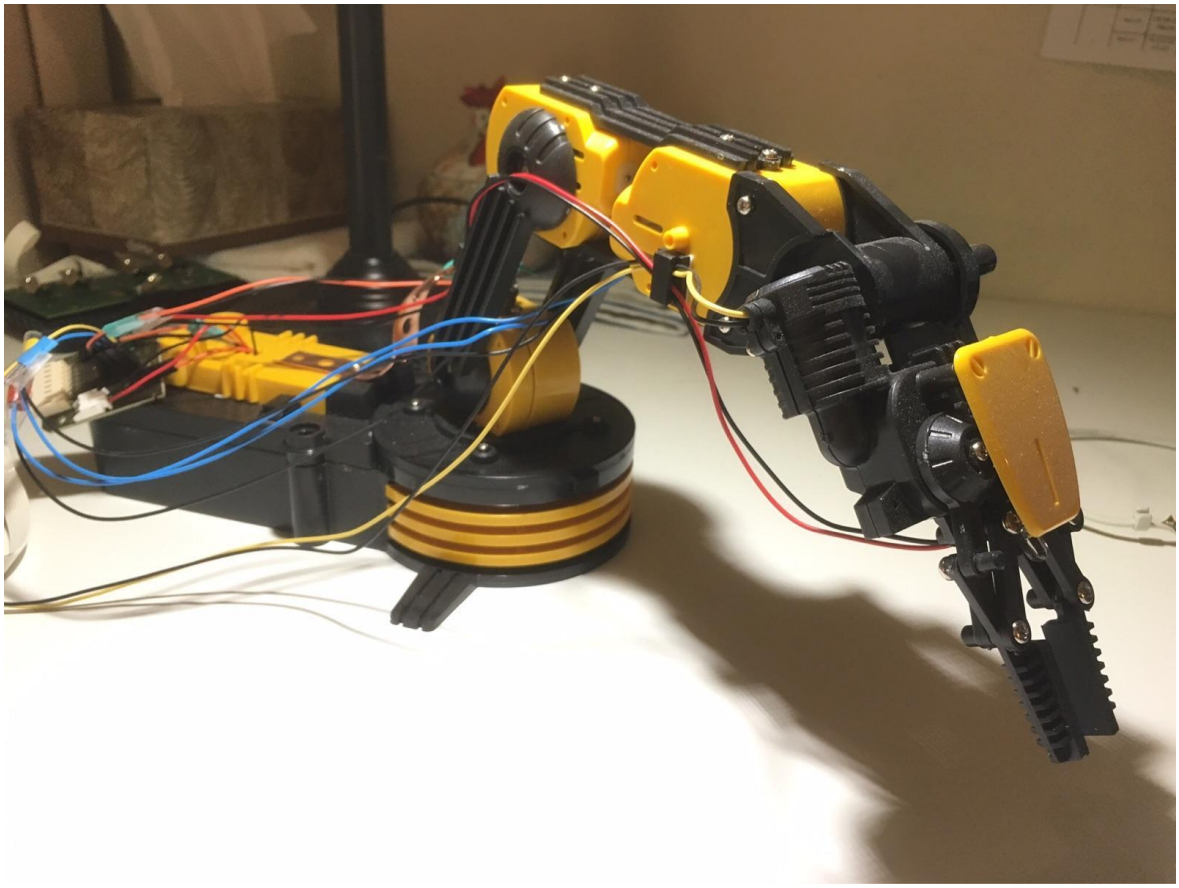

Figure 1.1: Robotic Arm Edge (Kinematic Linkage) with Five Joints

Unfortunately, they are quite complex, involving heavy data analysis and computing to specify its behavior in order to be pragmatic for the designer. But, in order to have a better appreciation and use for these “robot manipulators”, it is without uncertainty that learning how to specify the physical dynamics of the robotic system is a necessary step. So then, how are the physical dynamics of kinematic linkages best represented?

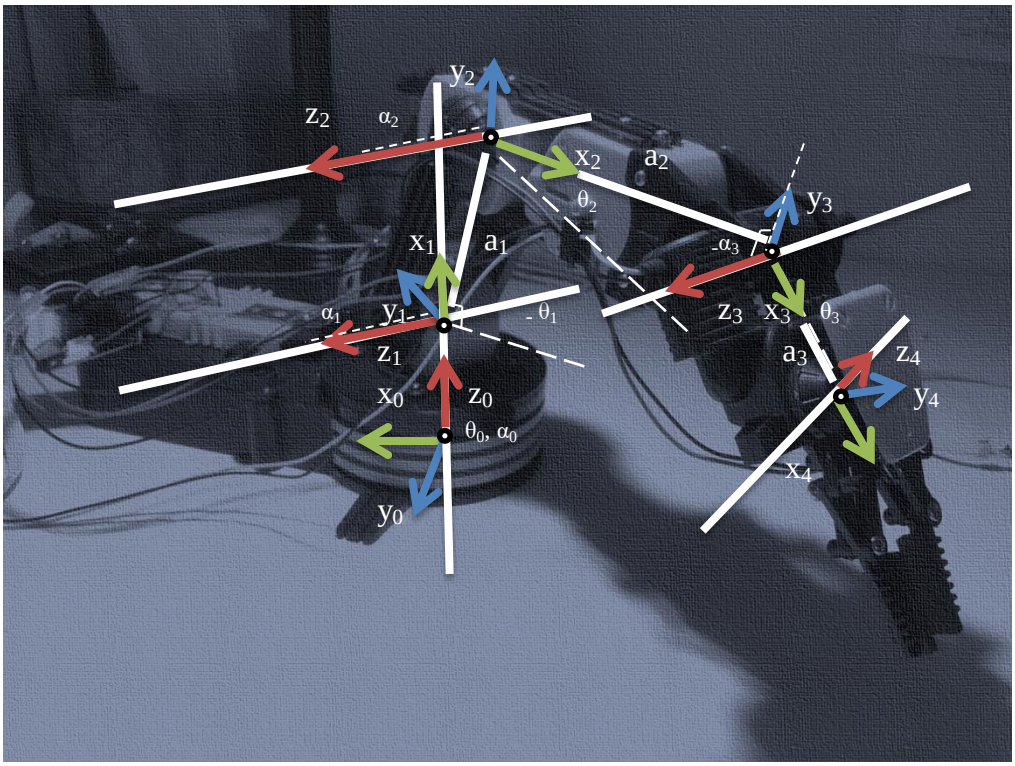

Figure 1.3: Simple Schematic of the Robotic Arm Edge

The physical dynamics of a kinematic linkage can best be described by forward kinematics, in which the transformation of the operational space parameters and the configuration space parameters define the behavior of the end effector's current position with respect to the world reference frame. This can be generated by using the Denavit-Hartenberg parameters, which can be approximated. Furthermore, in order to control the kinematic linkage to a desired position, the transformation found in the forward kinematic must be further used to observe its inverse kinematic.

Methods:

In order to compute the forward kinematics and implement inverse kinematics, the Denavit-Hartenberg parameters for the linkage of the robotic arm edge mentioned above have to be approximated.

These are the following Denavit-Hartenberg parameters:

- $\mathbf{a_i}$ is the distance between O_i and O_i'.
- $\boldsymbol{\alpha_i}$ is the angle between the z_{i+1} and z_i axes that are about the x_i axis (positive when rotating counter-clockwise).

- **d_i** is the distance between O_{i-1}' and $O_{i..}$
- **θ_i** is the angle between the x_{i+1} and the x_i about the z_i axis (positive when rotating counter-clockwise).

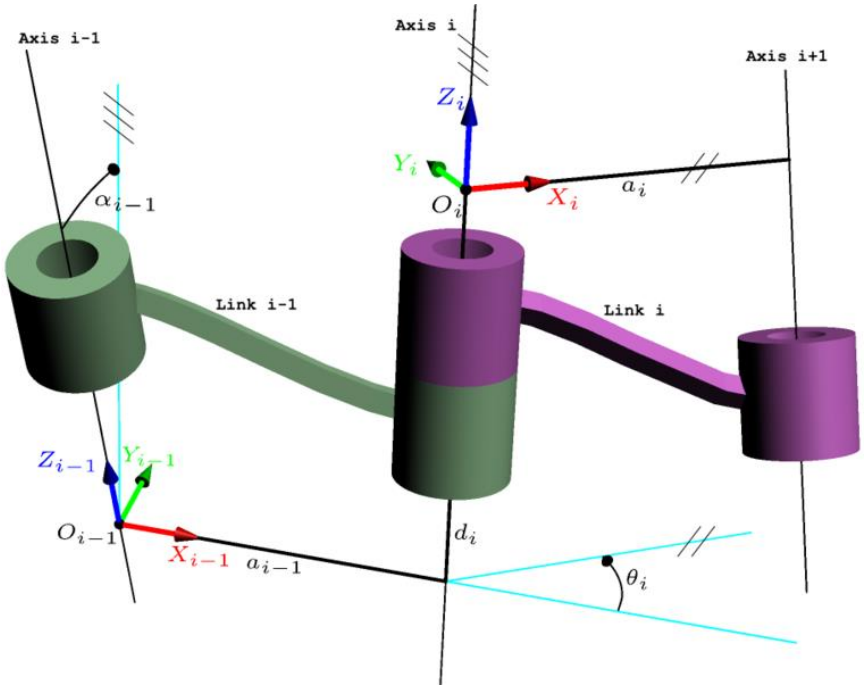

Figure 2.1: Modified D-H Parameters of a Kinematic Linkage [1]

Table 1.1: Approximated D-H Parameters for Robotic Arm Edge (Figure 1.1)

i joint(s)	**a_i (cm)**	**α_i (degrees)**	**d_i (cm)**	**θ_i (degrees)**
0	5	0	0	0
1	10	0	0	-90
2	10	0	0	45
3	10	-90	0	0

Forward Kinematics:

Using these following approximated parameters, the transformation matrix at the i^{th} joint relative to the $(i-1)^{th}$ joint frame can be generated using the following matrix:

$$T_i^{i-1} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_{i-1} \\ \sin\theta_i \cos\alpha_{i-1} & \cos\theta_i \cos\alpha_{i-1} & -\sin\alpha_{i-1} & \sin\alpha_{i-1} \cdot d_i \\ \sin\theta_i \sin\alpha_{i-1} & \cos\theta_i \sin\alpha_{i-1} & \cos\alpha_{i-1} & \cos\alpha_{i-1} \cdot d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Forward kinematic is a type of problem that uses the kinematic equation of a robot to analyze the current position of the end-effector for the values identified by the joint parameters relative to the

world frame. The same applies when finding the transformation matrix of the end effector with reference to the base frame, which is:

$$\boldsymbol{T_i^0 = T_1^0 T_2^0 T_3^0 \dots T_i^{i-1}}$$

Then, the kinematic problem is solved by calculating the homogeneous transformation vector to using the following equation:

$$\boldsymbol{p^{\sim 0} = T_i^0 p^{\sim n}}$$

The homogeneous transformation at the n frame is simply a generic position vector relative to the end-effector reference frame:

$$\boldsymbol{p^{\sim n} = \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix}}$$

Inverse Kinematics:

So long as the end-effector position is clearly fixed relative to the n^{th} frame, the end-effector position vector relative to the world frame is simply a function that is dependent on joint space state variables:

$$\boldsymbol{x = f(q)}$$

Using the following statement, the following equations can be determined:

$$\boldsymbol{dq = J_v^{-1}(q) \cdot dx}$$

$$\boldsymbol{\frac{dx}{dy} = J_v(q) \frac{dq}{dt}}$$

Note: $J_v(q)$ is represented as the Jacobian matrix as a function of q, which is the joint space vector.

This provides the linear velocity of the end-effector and for the angular velocity, we have:

$$\boldsymbol{\omega = \sum_{i=1}^{n} \epsilon_i z_i \dot{q}_i = J_\omega \dot{q}}$$

1. Given the operational space vector x_0 and joint space vector q_0, pick a step size δx towards the desired position
2. Use the equation, $\boldsymbol{\delta q = J_v^{-1}(q) \cdot \delta x}$, to calculate $\boldsymbol{q_{i+1} = q_i + \delta q}$
3. Calculate the corresponding operational space vector x_{i+1} using the forward kinematics method generated using MATLAB

4. See that the operational space vector is close to the desired point. If not, go back to step 1 with a new operational space vector x and a new joint space vector q until the desired joint space vector q is found.

   The following space vector q found in step 4 solves for the inverse kinematic problem.

   *** The MATLAB codes can be downloaded from https://github.com/jpark6694/EE-183DA-Foward-and-Inverse-Kinematic-for-Kinematic-Chains .

Results:

Using the MATLAB code function from the FK.m file, the following theta angles were inputted:

$$\boldsymbol{p\_0 = FK([0, -90, 45, 0])}$$

- The generated outputs were:

$$\boldsymbol{p^{\sim 0} = \begin{bmatrix} 8.26607 \\ 0 \\ -21.4881 \end{bmatrix}}$$

Figure 3.1: The Current Position Vector of the Robotic Arm Edge

The output is the position vector that the end vector is currently at with respect to the world frame.

Using the MATLAB code function from the IK.m file, the following position vector was inputted:

$$\boldsymbol{q = IK([8.26607;\quad 0;\quad -21.4881])}$$

- The generated outputs were:

$$\boldsymbol{q = \begin{bmatrix} 5.45377 \cdot 10^{-15} \\ 5.73511 \\ 5.73511 \\ 5.75162 \end{bmatrix}}$$

Figure 3.2: The Linkage Angles for the Desired Position

The resulting trajectory in Figure (3.1) shows a position vector of the robotic arm edge that has a non-zero element for more than one axes. Compared to a straight-line motion, a straight line motion only has a single non-zero element for the given axes in the position vector because a

prismatic joint only moves in the direction of a single axis. The resulting trajectory in Figure (3.2) shows the angles that each joint must be positioned at in order for the end-effector to reach a desired goal. The results in Figure (3.2) are the angles that will move the robot back to the desired position vector in Figure (3.1), which is true because the inverse kinematic simply converts the joint space vector back to the operational space vector. Forward kinematic does the vice-versa. Although the resulting angles for Figure (3.2) are different from the angles previously inputted for Figure (3.1), this may be because more than one generalized coordinate can give the desired end-effector position, depending on how the frames were oriented. For a straight-line linear motion, the generated output for Figure (3.2) would be empty because, in a prismatic motion, the robotic arm cannot make any revolute joint movements in order to form an angle. Furthermore, based on the methods above, a more desirable trajectory can be found by comparing the position vector that was outputted to a certain threshold in order to find the right generic homogenous transformation vector, which will minimize the error. Finally, this lab took me 24 hours to finish. All sections were equally challenging; however, the hardest is the introduction because of the sample task.

Reference(s):

[1] Wikipedia contributors, "*Denavit-Hartenberg parameters*" (November 16, 2016), https://en.wikipedia.org/wiki/Denavit%E2%80%93Hartenberg_parameters

[2] Prof. Ankur Mehta, "Lecture 2: Robot Description and State" (Fall 2016), https://ccle.ucla.edu/pluginfile.php/1669435/mod_resource/content/1/ee209as-lecture-notes.pdf

**Github Repository:
https://github.com/jpark6694/EE-183DA-Foward-and-Inverse-Kinematic-for-Kinematic-Chains .

*** For this lab, I worked with **Sumedh Vijay** on a common MATLAB code. I wrote the code for the forward kinematics and Sumedh did the inverse kinematics. The total percentage of the work done on my part is 45%, and Sumedh has done 55% of the work.