Forward Kinematics

Drawing a Robot Arm

Contents

1 Overview

In this assignment, you will write a program that draws a 3-D robot arm. While the arm will be drawn in 3-D (i.e., possibly using cylinders as parts and spheres as joints), the arm's movement will be restricted to a plane (i.e., its movement will happen on the xy-plane only). You will implement the forward-kinematics function of the robot arm, and try out a few poses given a set of joint angles of your choice.

The robot arm should have at least 4 parts. An example of a 4-part robot arm is shown in Figure 1. The base of the arm will be fixed (i.e., $\phi_1 = 0$ and the part does not rotate) and placed somewhere away from the origin of the global coordinate system. All other joints rotate within a 180-degree range. Each part has its own coordinate frame, and the frame's y-axis is aligned with the part.

These notes describe the main concepts and mathematical background needed to complete the assignment. Further reading might be needed to fill in some gaps.

2 Forward kinematics and drawing the arm

2.1 Forward kinematics of the robot arm

We want to define the function $\mathbf{e}$ that controls the arm's pose and the end-effector position (and pose), i.e., the forward-kinematics function. This function takes the vector of joint angles $\mathbf{\Phi}$ as input and returns the location of the end-effector, $\mathbf{e}(\mathbf{\Phi})$. Here, $\mathbf{\Phi} = (\phi_1, \phi_2, \ldots, \phi_M)^\mathsf{T}$ represent the M joint angles of the kinematic object and $\mathbf{e} = (e_1, e_2, \ldots, e_N)^\mathsf{T}$ represent the N degrees-of-freedom (DOF) of the end effector. The forward-kinematics function f computes the position of the end effector given the joint-angle configuration, i.e.:

$$\mathbf{e} = f(\mathbf{\Phi}). \tag{1}$$

In many applications, $\mathbf{e}$ has 6 DOFs, three for 3-D rotation, and three for its 3-D location. In this assignment, we will assume that we are interested only in the end-effector's position. As a result, $N = 3$ and $\mathbf{e} = (e_x, e_y, e_z)^\mathsf{T}$. Also, the robot arm in Figure 1 has 4 joint angles, i.e., $\mathbf{\Phi} = (\phi_1, \phi_2, \phi_3, \phi_4)^\mathsf{T}$.

The robot arm's parts and joints are illustrated in the diagram in Figure 1, from which we can derive the arm's forward-kinematic function $\mathbf{e}(\mathbf{\Phi})$. The base of the arm is a *cylindrical* joint (joint 1) that rotates about the z_0-axis. All other joints are *elbow* joints that rotate about their y-axis. We assume that each joint is centered at a local coordinate frame whose z-axis is aligned with the corresponding arm part that attaches to the joint. Figure 1(b) shows the 4 coordinate frames of the robot arm, and $x_0 y_0 z_0$ is the *world* coordinate frame. The joint center $\mathbf{p}_5$ is the end-effector position $\mathbf{e}$.

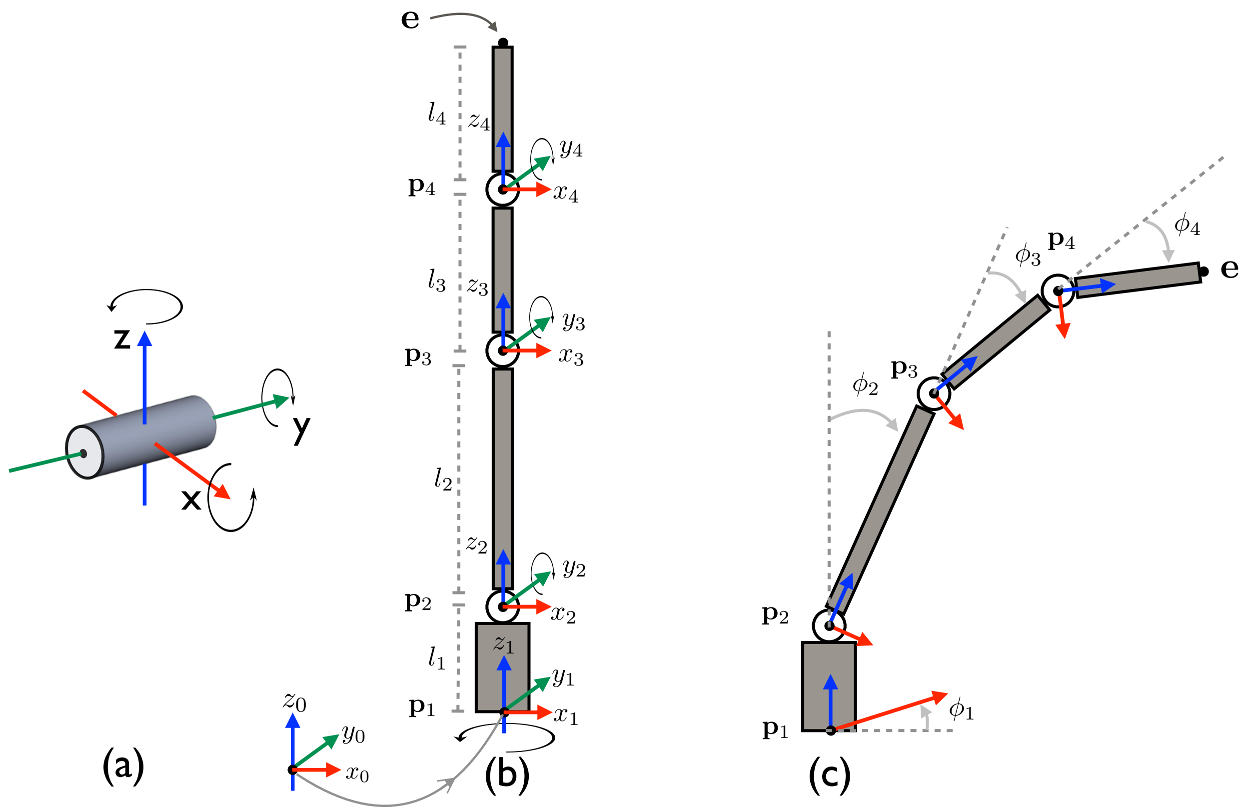


Figure 1: (a) Elbow join with y-axis rotation. The arm part that connects to the joint is aligned with that part's z-axis. (b) Robot arm in its home (i.e., zero) configuration. (c) Part l_1 rotates about the z_1-axis. All other parts have elbow joints and rotate about the part's y-axis. In this diagram the joint center $\mathbf{p}_5$ is the end-effector position $\mathbf{e}$.

To build the forward-kinematics equation, our first step is to build the individual change-of-coordinate transformations (i.e., rigid-body transformation) that convert points from frame i to frame $i+1$. The individual transformations are described next.

2.1.1 Individual coordinate-frame transformations

The first transformation converts coordinates from frame 0 (world frame) to frame 1. Frame 1 is given by a rotation about the z_0-axis followed by a translation by the location of the joint

$\mathbf{p}_1$ in world coordinates, which is given by:

$$T_{0,1} = \begin{bmatrix} R_z(\phi_1) & \mathbf{p}_1 \\ \mathbf{0}_{1x3} & 1 \end{bmatrix} = \begin{bmatrix} \cos\phi_1 & -\sin\phi_1 & 0 & 0 \\ \sin\phi_1 & \cos\phi_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2}$$

In the case of Equation 2, $\mathbf{p}_1 = (0,0,0)^{\mathsf{T}}$. This is the location of the joint center corresponding to the base of the arm. If we want the base of the arm to be elsewhere, we change the coordinates of $\mathbf{p}_1$ in the transformation matrix.

The transformations the next three subsequent coordinate frames of the arm are rotations about the part's y_i-axes, followed by the translation of size l_i along that part's z_i-axis, i.e.:

$$T_{1,2} = \begin{bmatrix} R_y(\phi_2) & \mathbf{t}_z(l_1) \\ \mathbf{0}_{1x3} & 1 \end{bmatrix} = \begin{bmatrix} \cos\phi_2 & 0 & \sin\phi_2 & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\phi_2 & 0 & \cos\phi_2 & l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3}$$

$$T_{2,3} = \begin{bmatrix} R_y(\phi_3) & \mathbf{t}_z(l_2) \\ \mathbf{0}_{1x3} & 1 \end{bmatrix} = \begin{bmatrix} \cos\phi_3 & 0 & \sin\phi_3 & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\phi_3 & 0 & \cos\phi_3 & l_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4}$$

and

$$T_{3,4} = \begin{bmatrix} R_y(\phi_4) & \mathbf{t}_z(l_3) \\ \mathbf{0}_{1x3} & 1 \end{bmatrix} = \begin{bmatrix} \cos\phi_4 & 0 & \sin\phi_4 & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\phi_4 & 0 & \cos\phi_4 & l_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{5}$$

Finally, the last transformation of the kinematic chain, T_{45}, takes us from frame 4 to the end-effector frame (i.e., frame 5). Because we decided that our end effector has no orientation in space, the rotation submatrix is just the identity matrix. It also means also the end-effector frame is aligned with frame 4. The final transformation is:

$$T_{4,5} = \begin{bmatrix} \mathbb{1}_{3x3} & \mathbf{t}_z(l_4) \\ \mathbf{0}_{1x3} & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & l_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{6}$$

Note that T_{45} takes us from frame 4 to the frame of the end effector (i.e., frame 5). Because we decided that our end effector has no orientation in space, the rotation submatrix is just the identity matrix. This means that the end-effector frame is aligned with frame 4.

2.1.2 Frames in world coordinates

The final mathematical step for creating the forward-kinematics equation is to represent each joint-frame in world coordinates. This representation is based on the fact that we can combine consecutive transformation to compose new transformations. For example, the coordinate frame 2, expressed in world coordinates is given by $T_{0,2} = T_{0,1}T_{1,2}$. The composition of transformations is illustrated in Figure 2.

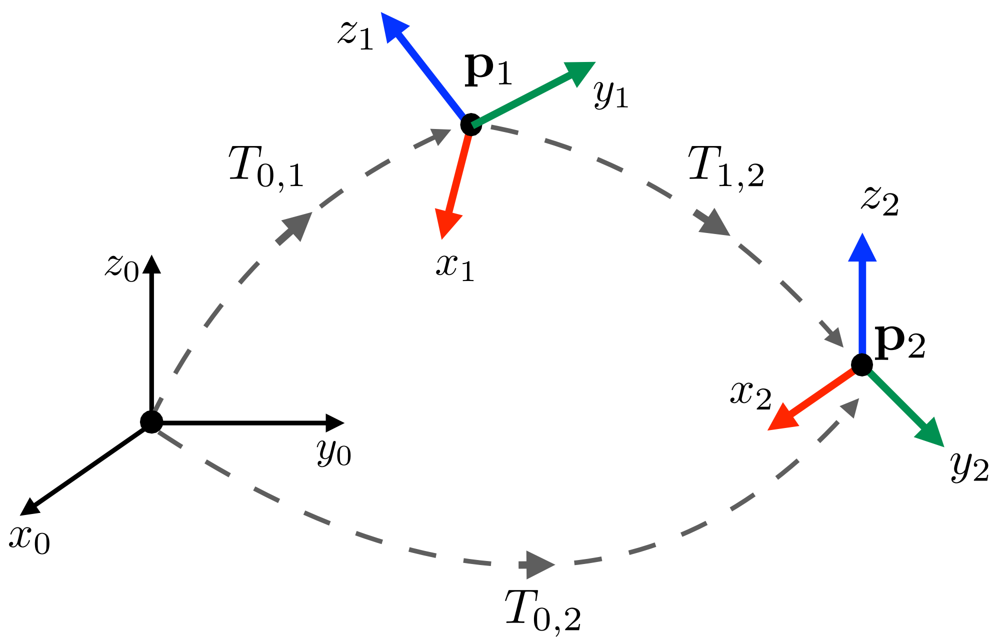


Figure 2: Coordinate frame 2 in world coordinates (frame 0) is given by a composed transformation formed by multiplying the matrices of the transformation chain, i.e., $T_{0,2} = T_{0,1}T_{1,2}$.

The coordinate frames at the joint points with respected to world coordinates are given by:

$$
\begin{aligned}
T_{0,1} &= \begin{bmatrix} R_{0,1} & \mathbf{p}_1 \\ \mathbf{0} & 1 \end{bmatrix}, \\
T_{0,2} &= T_{0,1}T_{1,2} = \begin{bmatrix} R_{0,2} & \mathbf{p}_2 \\ \mathbf{0} & 1 \end{bmatrix}, \\
T_{0,3} &= T_{0,2}T_{2,3} = T_{0,1}T_{1,2}T_{2,3} = \begin{bmatrix} R_{0,3} & \mathbf{p}_3 \\ \mathbf{0} & 1 \end{bmatrix}, \\
T_{0,4} &= T_{0,3}T_{3,4} = T_{0,1}T_{1,2}T_{2,3}T_{3,4} = \begin{bmatrix} R_{0,4} & \mathbf{p}_4 \\ \mathbf{0} & 1 \end{bmatrix}, \\
T_{0,5} &= T_{0,4}T_{4,5} = T_{0,1}T_{1,2}T_{2,3}T_{3,4}T_{4,5} = \begin{bmatrix} R_{0,5} & \mathbf{p}_5 \\ \mathbf{0} & 1 \end{bmatrix}.
\end{aligned} \tag{7}
$$

Finally, the joint center $\mathbf{p}_i$ of the robot arm is the translation component of the transformation T_{0i} (i.e., the elements in the first 3 rows of the the last column).

2.1.3 Forward-kinematics function

The implementation of the forward-kinematics function is listed in Algorithm 1.

Algorithm 1 Forward Kinematics

function forwardKinematics($\mathbf{\Phi}$)
  $l_1, \ldots, l_4 \leftarrow$ Lengths of each part of the arm
  $T_{i-1,i}$, for $i = 1, \ldots, 4$. ▷ Construct individual transformations (Eqs. 2 to 6)
  for $i = 2, \ldots, 5$ **do**
    $T_{0,i} \leftarrow T_{0,i-1} * T_{i-1,i}$ ▷ Compute frames in world coordinates (Eq. 7)
  end for
  return $(T_{0,1}, \ldots, T_{0,5})$ ▷ Return all frames w.r.t. the world frame
end function

The translation component of matrix $T_{0,i}$ is the location of the joint center of part i. Thus, the location of the end-effector is given by translation component of the last transformation matrix. Algorithm 2 lists the steps to obtain the end-effector location.

2.2 Drawing the robot arm

To draw the robot arm, we must draw each part at their associated joint center, $\mathbf{p}_i$. This location is the 3-D translation of the coordinate-frame transformation. Also, the part must

Algorithm 2 End-Effector Location

function $\mathbf{e}(\mathbf{\Phi})$
    $(T_{0,1}, \ldots, T_{0,N}) \leftarrow$ `forwardKinematics` $(\mathbf{\Phi})$ ▷ Obtain frames (Algo 1)
    $\mathbf{p} \leftarrow$ `lastColumnOf` $(T_{0,N})$
    return p ▷ Return (x,y,z) location of end effector
end function

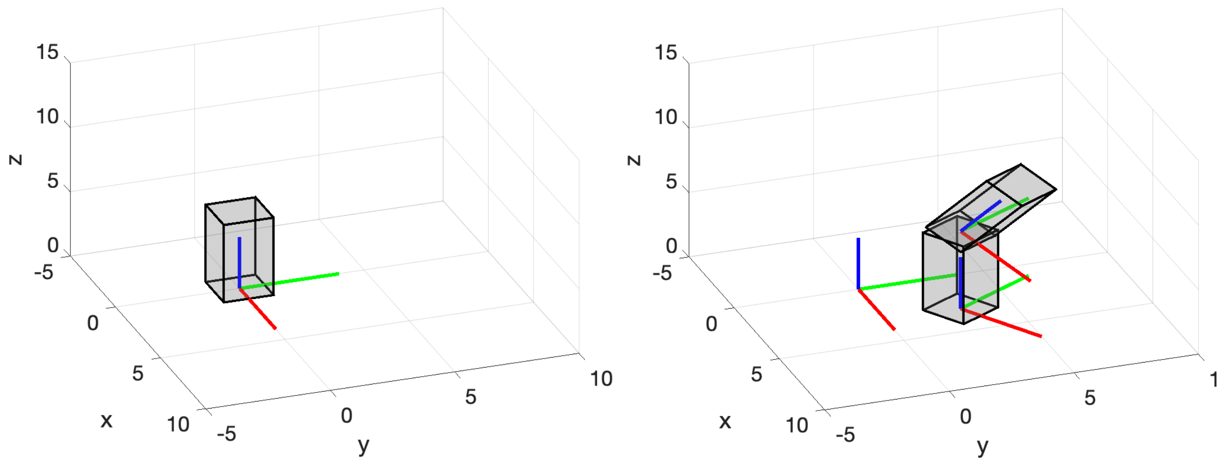
be rotated according to the part's coordinate-frame transformation. To place the part at the correct location and orientation, we transform each part using the transformations calculated in the forward-kinematics equations (Equation 7).

2.2.1 Example of drawing parts using coordinate-frame transformations

In this example, we draw two parts of an articulated object. Here, each part is represented by a rectangular prism of the same size. First, we need to construct the part, which will have the center of its base located at the origin of the world coordinate frame. Also, we want the frame's z-axis to be aligned with the part's longest side. The following set of vertices describe the rectangular prism shown in Figure 3(a).

$$v = \begin{bmatrix} -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 0 & 0 & 10 & 10 & 10 & 10 & 0 & 0 \end{bmatrix}. \tag{8}$$

We will assume that the object's base joint is located at $\mathbf{p}_1 = (3, 3, 0)^\mathsf{T}$ and object's part 1 rotates about the z-axis of frame 1. Let $T_{0,1}$ and $T_{0,2}$ be the coordinate transformations for frame 1 and frame 2. To draw each part in their correct place and orientation, we need only to transform the vertices of the prism by coordinate-frame matrices before we draw the part, e.g., `draw3D` $(T_{0,1} * v)$, `draw3D` $(T_{0,2} * v)$. Figure 3(b) shows the result of plotting the prism using transformations $T_{0,1}$ and $T_{0,2}$. Algorithm 3 summarizes the procedure to draw the robot parts.



(a) (b)

Figure 3: (a) A rectangular prism. The center of its base is located at the origin. The longest size of the prism is aligned with the z-axis of the world coordinate frame. (b) A two-part arm built with rectangular prisms.

Algorithm 3 Draw an N-part arm: example using a prism as part.

$V_1 \leftarrow$ Vertices of part 1 ▷ Graphical object describing part 1 (e.g., Eq. 8)

$V_2 \leftarrow$ Vertices of part 2

$\vdots$

$V_N \leftarrow$ Vertices of part N

$(T_{0,1}, \ldots, T_{0,N+1}) \leftarrow$ `forwardKinematics`$(\mathbf{\Phi})$ ▷ Obtain frames (Algo 1)

for $i = i, \ldots, N$ **do**

  $V_i' \leftarrow T_{0,i} * V_i$ ▷ Transform part's vertices to its local frame in Eq. 7

  `drawPart`(V_i') ▷ Draw the part

end for

A Rigid-body transformation

Rotations about x-axis, y-axis, and z-axis:

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \tag{9}$$

$$R_y(\phi) = \begin{bmatrix} \cos\phi & 0 & \sin\phi \\ 0 & 1 & 0 \\ -\sin\phi & 0 & \cos\phi \end{bmatrix} \tag{10}$$

$$R_z(\phi) = \begin{bmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{11}$$

Translation by a vector $\mathbf{t} = (t_x, t_y, t_z)^\mathsf{T}$. The rigid-body transformation in block-matrix notation:

$$T = \begin{bmatrix} r_1 & r_2 & r_3 & t_x \\ r_4 & r_5 & r_6 & t_y \\ r_7 & r_8 & r_9 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} R_{3\times3} & \mathbf{t}_{3\times1} \\ \mathbf{0}_{1x3} & 1 \end{bmatrix} \tag{12}$$

Example transformation with rotation about the z-axis:

$$T = \begin{bmatrix} \cos\phi & -\sin\phi & 0 & t_x \\ \sin\phi & \cos\phi & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{13}$$

References

[1] David E. Breen. Cost minimization for animated geometric models in computer graphics. *The Journal of Visualization and Computer Animation*, 8(4):201–220, 1997.