Forward Kinematics: A robotics classic

Alireza Azimi Veronika Ivanytska

2025-06-28

Introduction

Have you ever tried to pinpoint the exact position of your hand in 3D space just by looking at it? It's not so easy! In robotics, knowing the precise location of an **end-effector**—like a robot's hand—is essential for tasks such as picking up objects or assembling parts. This is where *forward kinematics* come in: it's the mathematical process that allows us to calculate the position and orientation of a robot's end-effector based on its joint angles and link lengths.

But how do we actually perform these calculations? One of the most widely used methods is the *Denavit-Hartenberg (DH) transformation*. By inputting the robot's joint angles and link dimensions, the DH transformation gives us the exact cartesian coordinates and orientation of the end-effector in space. This powerful technique forms the foundation for controlling and programming robotic arms in real-world applications.

Denavit-Hartenberg (DH) Transformation

$$T_i^{i+1} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \cos\alpha_i & \sin\theta_i \sin\alpha_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\theta_i \cos\alpha_i & -\cos\theta_i \sin\alpha_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Where the parameters are as follows:

- θ_i: Joint angle — rotation about the (z_{i-1}) axis
- d_i: Link offset — translation along the (z_{i-1}) axis
- a_i: Link length — translation along the (x_i) axis
- α_i: Link twist — rotation about the (x_i) axis

T_0^1 represents a transformation between frame 0 (the base) and frame 1 which is connected through link 1.

Example 1: One Link and One Angle

Let's warmup with the simplest example we can think of. When we have one link. This is quite simple and doesn't require the transformation as we can simply solve it using trigonometry:

- θ_i: 60 degrees ($\pi/3$)
- d_i: 0
- a_i: 1 m
- α_i: 0

$$T_0^1 = \begin{bmatrix} \cos\frac{\pi}{3} & -\sin\frac{\pi}{3} & 0 & \cos\frac{\pi}{3} \\ \sin\frac{\pi}{3} & \cos\frac{\pi}{3} & 0 & \sin\frac{\pi}{3} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & -\frac{\sqrt{3}}{2} & 0 & \frac{1}{2} \\ \frac{\sqrt{3}}{2} & \frac{1}{2} & 0 & \frac{\sqrt{3}}{2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The x,y coordinates of the end effector belong to $(T_0^1)_{14}$ and $(T_0^1)_{24}$ respectively.

Let's put this into code using `python`:

```python
import numpy as np
def dh_matrix(theta, d, a, alpha):
    """
    Computes the Denavit-Hartenberg (DH) transformation matrix.
    Parameters:
    - theta (float): The joint angle in radians.
    - d (float): The offset along the previous z-axis to the common normal.
    - a (float): The length of the common normal (distance between z-axes).
    - alpha (float): The angle in radians between the previous z-axis and the current z-axis

    Returns:
    - numpy.ndarray: A 4x4 transformation matrix representing the DH parameters.
    """
    c_theta, s_theta = np.cos(theta), np.sin(theta)
    c_alpha, s_alpha = np.cos(alpha), np.sin(alpha)
    return np.array([
        [c_theta, -s_theta * c_alpha, s_theta * s_alpha, a * c_theta],
        [s_theta, c_theta * c_alpha, -c_theta * s_alpha, a * s_theta],
```

```
        [0,  s_alpha,      c_alpha,       d],
        [0,  0,       0,        1]
    ])
```

Now let's test our function on our example:

```
T_0 = dh_matrix(np.deg2rad(60), 0, 1, 0)
T_0
```

```
array([[ 0.5      , -0.8660254,  0.       ,  0.5      ],
       [ 0.8660254,  0.5      , -0.       ,  0.8660254],
       [ 0.       ,  0.       ,  1.       ,  0.       ],
       [ 0.       ,  0.       ,  0.       ,  1.       ]])
```

Example 2: Extend to 2 Links

Link 1:

- θ_i: 60 degrees ($\pi/3$)
- d_i: 0
- a_i: 1 m
- α_i: 0

Link 2:

- θ_i: 30 degrees ($\pi/3$)
- d_i: 0
- a_i: 1 m
- α_i: 0

```
T_0 = dh_matrix(np.deg2rad(60), 0, 1, 0)
T_1 = dh_matrix(np.deg2rad(-30), 0, 1, 0)

## Chain together
T = T_0 @ T_1
T
```

```
array([[ 0.8660254, -0.5      ,  0.       ,  1.3660254],
       [ 0.5      ,  0.8660254,  0.       ,  1.3660254],
       [ 0.       ,  0.       ,  1.       ,  0.       ],
       [ 0.       ,  0.       ,  0.       ,  1.       ]])
```

Explore

A good exercise would be to extend the tranformation to 3 dimensions. Have fun!