2D Kinematics

Consider a robotic arm. We can send it commands like, move that joint so it bends at an angle $\theta$. Once we’ve set each joint, that’s all well and good. More interesting, though, is the question of once we set these angles, where is the end of the robotic hand? This is kinematics. We’ll also look at the problem of inverse kinematics, where we want the end of the hand to be at a certain point, and so need to figure out what angles to set the joints to.

Before we move on to the full arm, with all of its possible degrees of freedom, it is easier to consider a two-dimensional arm like the following, where our goal is to find at which point in Cartesian XY plane the end of the arm is located.

![Figure 1: 2D arm](image1)

Now, suppose that we know the following:

- We set the first joint to rise from the X-axis at an angle of $\pi/4$ radians (the red angle in Figure 2, and the first link (the name for the part of the arm connecting two joints) is 5 meters long.
- We set the second joint to be $-\pi/6$ radians from straight (the blue angle in Figure 2), and the second link is 3 meters long.

![Figure 2: 2D arm with labelled angles](image2)

Notice that on the second joint, our information is relative from the location of the first link. So, where is the end of the arm? Your first solution may involve trigonometry, so let’s do that.

The second joint is located at $(5 \cdot \cos(\pi/4), 5 \cdot \sin(\pi/4))$. That’s easy enough. This means the end of the arm is at $(5 \cdot \cos(\pi/4) + 3 \cdot \cos(\pi/4 - \pi/6), 5 \cdot \sin(\pi/4) + 3 \cdot \sin(\pi/4 - \pi/6))$, or about $(6.4, 4.3)$. And that’s right, it works! But, perhaps you can see why we won’t want to do this for a more complicated arm.
You have to be constantly keeping track of the total angles (like when we had to do $\pi/4 - \pi/6$ above). Add in another dimension, and the trigonometric solution would get very hairy indeed. So let’s try something else.

**Matrix Addition and Multiplication**

To get this done, we’re going to need some very basic linear algebra, which we’ve seen before. First, note that to add (or subtract) two matrices, they must be the same size and shape. This is because matrix addition (and subtraction) is merely piecewise:

$$\begin{pmatrix} a & b \\ c & d \\ e & f \end{pmatrix} + \begin{pmatrix} g & h \\ i & j \\ k & l \end{pmatrix} = \begin{pmatrix} a + g & b + h \\ c + i & d + j \\ e + k & f + l \end{pmatrix}.$$  

So, if they’re not the same size, addition or subtraction just doesn’t make any sense.

Before we talk about matrix multiplication, let’s define how we talk about matrix sizes. If a matrix $M$ is $m \times n$, that means it has $m$ rows, and $n$ columns.

Now, if $A$ and $B$ are matrices, and we want to perform the multiplication $AB$, then the number of columns in $A$ and the number of rows of $B$ must match. In other words, if $A$ is $m \times n$, then $B$ must be $n \times p$. The result of this multiplication would be $m \times p$, where the $i$th row and the $j$th column are the dot product of the $i$th row of $A$ and the $j$th column of $B$. Consider, for example, the following multiplication:

$$\begin{pmatrix} a & b & c \\ d & e & f \end{pmatrix} \begin{pmatrix} g \\ h \\ i \end{pmatrix} = \begin{pmatrix} ag + bh + ci \\ dg + eh + fi \end{pmatrix}.$$  

**Transformational Matrices**

Now the thing that’s going to make our life easier is we can represent a point $(x, y)$ in some coordinate frame as a vector, $\begin{pmatrix} x \\ y \end{pmatrix}$. It turns out that we can move this point around in our space by using transformation matrices.

**Scaling**  Suppose we have some point $(x, y)$, and we’d like to move it so that the $x$-coordinate is five times bigger than it was before. What matrix can we use to perform this transformation? If you look at how matrices are multiplied, hopefully it’s clear this can be accomplished with

$$\begin{pmatrix} 5 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 5x + 0y \\ 0x + 1y \end{pmatrix} = \begin{pmatrix} 5x \\ y \end{pmatrix}.$$  

To expand this a bit, if you’d like to scale $x$ by some factor $a$, and $y$ by some factor $b$, you need the following transformation:

$$\begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} ax \\ by \end{pmatrix}.$$  

**Rotation**  Now suppose we have some point $(x_0, y_0)$, and we would like to perform a rotation around the origin by some angle $\phi$, like in Figure 3.

It’s not immediately clear how to change $\begin{pmatrix} x_0 \\ y_0 \end{pmatrix}$ into $\begin{pmatrix} x_1 \\ y_1 \end{pmatrix}$. So let’s just start writing down things that we know, and see if we can solve for $x_1$ and $y_1$. We’ll define $d$ to be the distance from the origin for both points (since it’s a rotation, it will be the same).
Figure 3: Rotating a point around the origin $\phi$ radians

\[
\begin{align*}
\cos(\theta) &= \frac{x_0}{d} \\
\sin(\theta) &= \frac{y_0}{d} \\
\cos(\phi + \theta) &= \frac{x_1}{d} \\
\sin(\phi + \theta) &= \frac{y_1}{d} \\
\cos(\phi + \theta) &= \cos(\phi) \cos(\theta) - \sin(\phi) \sin(\theta) \\
\sin(\phi + \theta) &= \sin(\phi) \cos(\theta) + \cos(\phi) \sin(\theta)
\end{align*}
\]

OK, so 1 and 2 are true just because that’s how we convert the polar coordinates $(\theta, d)$ to the Cartesian coordinates $(x_0, y_0)$. 3 and 4 are similar. 5 and 6 are just trigonometric identities.

So now we can set 3 equal to 5, and set 4 equal to 6, and then plug in 1 and 2 wherever possible.

\[
\begin{align*}
\frac{x_1}{d} &= \cos(\phi) \cos(\theta) - \sin(\phi) \sin(\theta) \\
\frac{y_1}{d} &= \sin(\phi) \cos(\theta) - \cos(\phi) \sin(\theta) \\
\frac{x_1}{d} &= \cos(\phi) \frac{x_0}{d} - \sin(\phi) \frac{y_0}{d} \\
\frac{y_1}{d} &= \sin(\phi) \frac{x_0}{d} + \cos(\phi) \frac{y_0}{d}
\end{align*}
\]

Multiply both sides by $d$...

\[
\begin{align*}
x_1 &= \cos(\phi) x_0 - \sin(\phi) y_0 \\
y_1 &= \sin(\phi) x_0 + \cos(\phi) y_0
\end{align*}
\]

And we’ve got it! That, we can do as a matrix. To rotate by $\phi$,

\[
\begin{bmatrix}
\cos(\phi) & -\sin(\phi) \\
\sin(\phi) & \cos(\phi)
\end{bmatrix}
\begin{bmatrix}
x_0 \\
y_0
\end{bmatrix} =
\begin{bmatrix}
x_1 \\
y_1
\end{bmatrix}
\]

**Translation** Translating by some distance (not scaling, but just picking it up and moving it some distance in some direction) requires a little bit different intuition. No $2 \times 2$ matrix exists that will perform this translation, since our only tool is multiplication, and we need to do addition.

The key is to use Homogeneous Matrices, and add a dimension to our points. To move our point in the $x$ direction by $m$, and in the $y$ direction by $n$, we multiply as follows:

\[
\begin{bmatrix}
1 & 0 & m \\
0 & 1 & n \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
x_0 \\
y_0 \\
1
\end{bmatrix} =
\begin{bmatrix}
x_0 + m \\
y_0 + n \\
1
\end{bmatrix}
\]
Translation and Rotation Combined If we want to perform a single transformation, in which we both translate and rotate, we can combine these into a single matrix:

\[
\begin{bmatrix}
\cos(\phi) & -\sin(\phi) & m \\
\sin(\phi) & \cos(\phi) & n \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
x_0 \\
y_0
\end{bmatrix}
=
\begin{bmatrix}
\cos(\phi)x_0 - \sin(\phi)y_0 + m \\
\sin(\phi)x_0 + \cos(\phi)y_0 + n \\
1
\end{bmatrix}
\]

We will rarely do this explicitly, but it will be useful to refer to later.

2D Kinematics using Transformations

Let’s consider the rotational transformation of Figure 3. One way to look at this is that we moved the point within the axes. Another way to look at it is that we had two different points, and we found a new set of axes such that the coordinates remained the same, but the axes were different between the two points. In other words, \( x_1 = x_0 \), but on different axes (red axes vs. black axes in the figure 4).

This is what we will do for our arm; we will rotate and translate the axes, so that our final point is at the origin. So, we’ll take our reference frame, or the original black axes, and rotate it \( \pi/4 \), creating a new frame \( N_1 \). We’ll then translate along that frame’s x-axis 5 meters, creating a new frame \( N_2 \). We’ll then rotate \( -\pi/6 \) radians, creating \( N_3 \). We’ll then translate along the x-axis to the end of the arm, creating \( N_4 \). These can be seen in Figure 5.
The matrix transformations to do this, therefore, are:

$$\begin{bmatrix}
\cos(\pi/4) & -\sin(\pi/4) & 0 \\
\sin(\pi/4) & \cos(\pi/4) & 0 \\
0 & 0 & 1
\end{bmatrix} \times
\begin{bmatrix}
1 & 0 & 5 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix} \times
\begin{bmatrix}
\cos(-\pi/6) & -\sin(\pi/6) & 0 \\
\sin(\pi/6) & \cos(\pi/6) & 0 \\
0 & 0 & 1
\end{bmatrix} \times
\begin{bmatrix}
1 & 0 & 3 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix} \times
\begin{bmatrix}
0 & 0 & 1
\end{bmatrix}$$

- Rotate to $N_1(T_1)$
- Translate to $N_2(T_2)$
- Rotate to $N_3(T_3)$
- Translate to $N_4(T_4)$
- Starting at the origin

When you perform all this multiplication, you get $\begin{bmatrix} 6.4 \\ 4.3 \\ 1 \end{bmatrix}$, or, the same location in the reference frame as we got from our trigonometry.

You’ll notice our notation, $A^T B$, which is how we notate a transition matrix from frame A to frame B. $R_{T_4} = R_{T_1} T_2 T_3 T_4$.

It turns out all of this is much, much easier than trying to do trig, because everything is relative to where you were before. In our third transformation matrix above, we didn’t need to know that before, we had already rotated $\pi/4$ radians; we only needed to know that now, we rotate down $\pi/6$ from wherever it is we already are. This concept keeps things much simpler with 3D arms.

**Calculating Orientation**

The above details how to find the position of the endpoint of the arm. But, what if we are also interested in the orientation? In 3D, this will mean the roll, pitch, and yaw of the end of the arm. In 2D, this means only the angle from the X axis in the reference frame.

This is remarkably easy. We note that our matrix $R_{T_4}$ in the above example, is itself a general transformation matrix from the reference frame $R$ to the frame $N_4$, of the form discussed in the paragraph “Translation and Rotation Combined.”

So, whatever is in the [0, 0] spot of that matrix, is the cosine of our total rotation $\psi$. So,

$$\psi = \cos^{-1}(R_{T_4}[0,0]).$$

With a perfect computer, with no round-off or precision errors, this would be exactly correct. Unfortunately... those don’t exist.

The problem comes from the fact that arccos and arcsin are pretty volatile. As $x$ approaches 1 or -1, small changes in $x$ (like from round-off errors) result in huge changes in $\arccos(x)$ and $\arcsin(x)$. So, just using one of those on their own is out.

What’s left, of course, is arctan, which is fortunately much more stable. If you remember your SOHCAHTOA, $\tan(\psi) = \frac{y}{x}$, so $\psi = \tan^{-1}(\frac{y}{x})$. But, note that because the single fraction is entered, we don’t know which quadrant we’re in. If $x$ and $y$ are both positive, (meaning $\psi$ should be between 0 and $\pi/2$) we’ll get the same answer as if $x$ and $y$ are both negative (meaning $\psi$ should be between $-\pi$ and $-\pi/2$).

OK. Fine. For exactly this reason, we won’t use arctan($x$). Instead, we’ll use a commonly-provided arctan2($y, x$). Notice this function takes both arguments, meaning the function knows which quadrant we’re supposed to be in.

OK! So, arctan2 will give us a stable answer in the correct quadrant... but what do we enter for its two parameters $x$ and $y$? It’s not the final point’s $x$ and $y$, as that gives us the angle from the $x$ axis, not the orientation of the final arm segment.

Remember that $\tan(\psi) = \frac{\sin(\psi)}{\cos(\psi)}$... and those two things we have $(\sin(\psi) = R_{T_4}[1,0]$ and $\cos(\psi) = R_{T_4}[0,0])$. So, our final, stable answer in the correct quadrant is:

$$\psi = \arctan2(R_{T_4}[1,0], R_{T_4}[0,0]).$$

Huzzah!