Forward and inverse kinematics for the 2-link manipulator (Two-Dimensional Kinematics)

Jaesung Oh [1], Yoonsung Joshua Ryu [2], Christi Kim [3], Yeji Cho [4]

[1] PhD Candidate, Hubo Lab, KAIST Korea Advanced Institute of Science and Technology, Seoul South Korea
[2] Yongsan International School of Seoul, Seoul, South Korea
[3] The Bronx High School of Science, Bronx NY
[4] Hunter College Campus School, New York, NY
yejicho3800@gmail.com

Abstract: The simplest of all the robots that can perform tasks in 2D kinematics is the 2-link manipulator, which has two-joint axles. One method of expressing the position of the 2-link manipulator is in joint space, which expresses the position of each joint. The other method uses coordinate data in Cartesian space (X, Y). In forward kinematics, data can be collected in the Cartesian space by using joint space; in inverse kinematics, the reverse is true.

[Jaesung Oh, Yoonsung Joshua Ryu, Christi Kim, Yeji Cho . **Forward and inverse kinematics for the 2-link manipulator (Two-Dimensional Kinematics).** *Researcher* 2017;9(11):36-39]. ISSN 1553-9865 (print); ISSN 2163-8950 (online). http://www.sciencepub.net/researcher. 5. doi:10.7537/marsrsj091117.05.

Keywords: robots: kinematics; manipulator

The following diagram and formulas refer to forward kinematics:

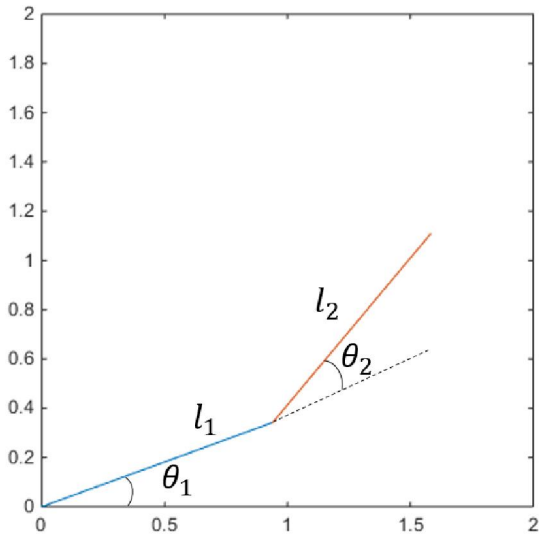

Figure 1. 2-link manipulator

$$x = l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2) \quad (1)$$
$$y = l_1 \sin(\theta_1) + l_2 \sin(\theta_1 + \theta_2) \quad (2)$$

The following formula allows for Analytic inverse kinematics with the 2-link manipulator:

$$x^2 + y^2 = l_1^2 + l_2^2 - 2l_{1_{l2}} \cos(\pi - \theta_2) \quad (3)$$

In formula 3, x and y are given, and using cosine, θ_2 can be calculated. Rearranging the equation, the following is obtained:

$$\cos(\theta_2) = \frac{x^2+y^2-l_1^2-l_2^2}{2l_1l_2} \quad (4)$$

$$\tan^2\left(\frac{\theta_2}{2}\right) = \frac{1-\cos(\theta_2)}{1+\cos(\theta_2)} = \frac{(l_1^2+l_2^2)^2-(x^2+y^2)}{(x^2+y^2)-(l_1^2-l_2^2)^2} \quad (5)$$

$$\theta_2 = \pm 2\tan^{-1}\left(\sqrt{\frac{(l_1^2+l_2^2)^2-(x^2+y^2)}{(x^2+y^2)-(l_1^2-l_2^2)^2}}\right) \quad (6)$$

After θ_2 is obtained using formula 6, θ_1 can be obtained using formula 7.

$$\theta_1 = \tan^{-1}\left(\frac{y}{x}\right) - \tan^{-1}\frac{l_2\sin(\theta_2)}{l_1+l_2\cos(\theta_2)} \quad (7)$$

There are several methods of performing inverse kinematics with the 2-link manipulator. Thus, the most appropriate formula for the application must be selected. By choosing the best formula for the 2-link manipulator's current configuration, any discontinuity of movement can be prevented.

A simulation created by using MATLAB can verify the kinematics and inverse kinematics of the 2-link manipulator presented above.

The initial conditions for the simulation are set as follows:

$$l_1 = 1\,[m], l_2 = 1\,[m], \theta_1 = 0\,[deg], \theta_2 = 0\,[deg]$$

Using these conditions, the position of the end-effector of the manipulator, using kinematics, can be confirmed as [x, y] = [2,0].

If the final desired position is [x, y] = [0, 1], the

manipulator's end-effector must be moved from 0 to 1 on the x-axis and from 2 to 0 on the y-axis. In the case of a real manipulator, there is a physical limitation of the actuator, so the manipulator cannot be moved from the current position to the target position in one step. Therefore, in the simulation, the manipulator must be operated by dividing the movement into 10 steps. In order to ensure the continuity of the motion, the target position is interpolated using the following function:

$$x[k] = x_s + \frac{\left(1-\cos\left(\frac{k}{10}\pi\right)\right)}{2}(x_d - x_s) \quad (8)$$

$$y[k] = y_s + \frac{\left(1-\cos\left(\frac{k}{10}\pi\right)\right)}{2}(y_d - y_s) \quad (9)$$

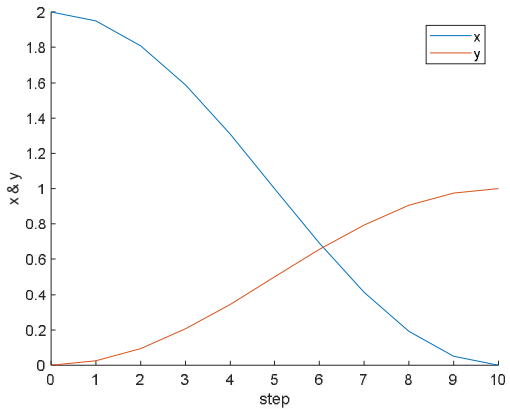

Figure 2. The desired movement from the starting position to the target position obtained by interpolation.

If the target position in the k-th step is (x [k], y [k]), the final trajectory of the manipulator can be obtained through inverse kinematics as follows:

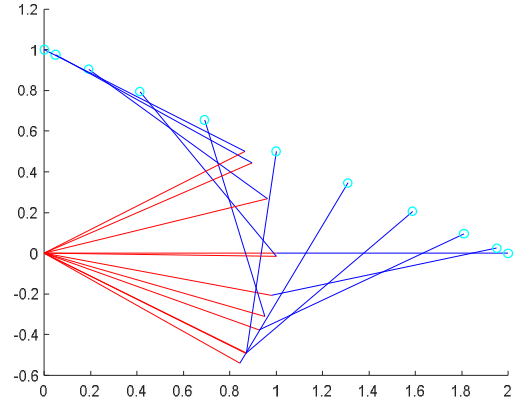

Figure 3. Moving trajectory of the manipulator using inverse kinematics.

There are numerical inverse kinematics in addition to the analytic inverse kinematics summarized above for handling inverse kinematics. Numerical inverse kinematics is based on Jacobian.

Jacobian describes the effect of movement in the joint space of a manipulator in a workspace by means of partial derivatives, and can be rearranged by differentiating the expression of forward kinematics. In other words, if x and y according to the joint angles are partially differentiated based on the respective joint values, the following can be summarized.

$$\frac{\partial x}{\partial q_1} = -l_1 \sin(\theta_1) - l_2 \sin(\theta_1 + \theta_2)$$

$$\frac{\partial x}{\partial q_2} = -l_2 \sin(\theta_1 + \theta_2)$$

$$\frac{\partial y}{\partial q_1} = l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2)$$

$$\frac{\partial y}{\partial q_2} = l_2 \cos(\theta_1 + \theta_2)$$

At this time, Jacobian is named as J and summarized as follows.

$$\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix} = J(\theta_1, \theta_2)\begin{bmatrix}\theta_1\\ \theta_2\end{bmatrix}$$

$$= \begin{bmatrix} -l_1 \sin(\theta_1) - l_2 \sin(\theta_1 + \theta_2) & -l_2 \sin(\theta_1 + \theta_2) \\ l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2) & l_2 \cos(\theta_1 + \theta_2) \end{bmatrix}\begin{bmatrix}\dot{\theta}_1\\ \dot{\theta}_2\end{bmatrix}$$

Considering the above equation, if we express the velocity to move from the current position to the target position by and, the velocity satisfying the equation can be obtained using Jacobian's Inverse, which is shown below.

$$\begin{bmatrix}\dot{\theta}_1\\ \dot{\theta}_2\end{bmatrix} = J^{-1}\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix}$$

However, at this time, the angle of the joint suggests the direction to be changed, not the absolute value. Therefore, it is necessary to iteratively perform the process of updating the joint angle by multiplying the appropriate gain to converge the final joint angle.

A simple algorithm is shown below.

Algorithm 1.

$x_t \leftarrow x_s$
$x_t \leftarrow x_d$

$q_1 \leftarrow 0$
$q_2 \leftarrow 0$

for k = 0 to 10

while (1)

$\dot{x} \leftarrow x[k]-x_t$
$\dot{y} \leftarrow y[k]-y_t$

$\begin{bmatrix} \dot{q_1} \\ \dot{q_2} \end{bmatrix} \leftarrow J(q_1, q_2)^{-1} \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix}$

$q_1 \leftarrow q_1 + K\dot{q}_1$ // K is the gain
$q_2 \leftarrow q_2 + K\dot{q}_2$

$\begin{bmatrix} x_t \\ y_t \end{bmatrix} \leftarrow F.K(q_1, q_2)$ // F.K is forward kinematics of the manipulator

if $((x[k] - x_t)^2 + (y[k] - y_t)^2 < \epsilon)$ // condition for termination
break
end if

end while

end for

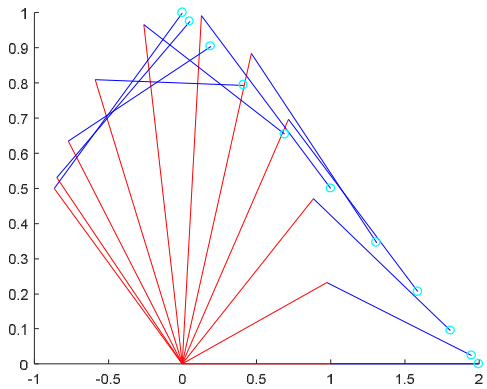

Figure 4. Numerical inverse kinematics Movement trajectory of a manipulator usage

Conclusion

This research was done with a motive to understand forward kinematics and inverse kinematics, the basics of robotics.

By using forward kinematics, information such as the manipulator's joint position and length of the link can be used to determine information such as the position and angle of the end-effector.

Through the use of inverse kinematics, movement in the joint space can be determined when work is given in the space. Most of the work that requires the manipulator is given within the Cartesian space, and thus becomes an essential part to the robot's movement. Inverse kinematics can be approached by two ways: an analytical approach and a numerical approach.

The analytical approach is typically used when the degree of freedom of the Cartesian space and the degree of freedom of the joint space are the same. Here, there may be multiple answers depending on the manipulator's structural characteristics. For this reason, the user must choose the answer.

The numerical approach is typically used when the degree of freedom of the Cartesian space and the degree of freedom of the joint space are different, and an iteration process is needed in order to accept the answer. The formula of forward kinematics can be replaced with simply a partial derivative, without need to use various mathematical calculations to find the answer. This method can thus be implemented into various types of systems.

Simulations of forward kinematics and inverse kinematics were processed through MATLAB Simulation, which was used to visually confirm the manipulator's movements.

Appendix – MATLAB code

```
close all
clear all
clc

q1 = 0;
q2 = 0;
l1 = 1;
l2 = 1;


k = 0:10;

xs = 2;
ys = 0;

xd = 0;
yd = 1;

x = xs + (1-cos (k/10*pi))/2*(xd-xs);
y = ys + (1-cos (k/10*pi))/2*(yd-ys);

figure (1)
hold on
```

```
plot (k,x,k,y)
xlabel ('step')
ylabel ('x & y')
legend ('x','y')



for k = 1:1:11

q2 = 2*atan (sqrt ( ((l1^2+l2^2)^2 - (x (k)^2 + y (k)^2)) / ((x (k)^2+y (k)^2)-(l1^2-l2^2)^2 ) ));
q1 = atan (y (k)/x (k)) - atan (l2*sin (q2)/(l1+l2*cos (q2)));

link1x = [0 l1*cos (q1)];
link1y = [0 l1*sin (q1)];

link2x = [l1*cos (q1) l1*cos (q1) + l2*cos (q1+q2)];
link2y = [l1*sin (q1) l1*sin (q1) + l2*sin (q1+q2)];

figure (2);
hold on

plot (link1x,link1y,'r')
plot (link2x,link2y,'b')
plot (l1*cos (q1) +

l2*cos (q1+q2),l1*sin (q1) + l2*sin (q1+q2),'oc')


end



q1 = 0;
q2 = 0;

tempx = 2;
tempy = 0;

for i = 1:11

while (1)

J = [-l1*sin (q1) -l1*sin (q1)-l2*sin (q1+q2); l1*cos (q1) l1*cos (q1) + l2*cos (q1+q2)]

xdot = x (i)-tempx;
ydot = y (i)-tempy;

qdot = pinv (J)*[xdot; ydot]/10;


q1 = q1+qdot (1);
q2 = q2+qdot (2);

tempx = l1*cos (q1) + l2*cos (q1+q2);
tempy = l1*sin (q1) + l2*sin (q1+q2);

if (norm ([x (i)- tempx, y (i)-tempy]) < 0.0001)
            break;
    end

end

link1x = [0 l1*cos (q1)];
link1y = [0 l1*sin (q1)];

link2x = [l1*cos (q1) l1*cos (q1) + l2*cos (q1+q2)];
link2y = [l1*sin (q1) l1*sin (q1) + l2*sin (q1+q2)];

figure (3);
hold on

plot (link1x,link1y,'r')
plot (link2x,link2y,'b')
plot (l1*cos (q1) + l2*cos (q1+q2),l1*sin (q1) + l2*sin (q1+q2),'oc')

end
```

Reference

1. Craig, John J. *Introduction to robotics: mechanics and control*. Vol. 3. Upper Saddle River: Pearson Prentice Hall, 2005.
2. Sciavicco, Lorenzo, and Bruno Siciliano. *Modelling and control of robot manipulators*. Springer Science & Business Media, 2012.

10/29/2017