

KIRA Inverse kinematics algorithm

Thursday, November 4, 2021 7:44 PM

1. Set the robot in an initial home state in the code / set ^{joint} angle variables to: Joint 0 = 0
 $\begin{matrix} \text{" 1 = 180} \\ \text{" 2 = 180} \end{matrix}$
2. Begin loop to get end-effector to goal point - (set $k=0$) within a certain tolerance

1. Create linear equation with goal point and point of joint k ,
 (goal is to move joint k some angle that will get the end effector to a point in this line).

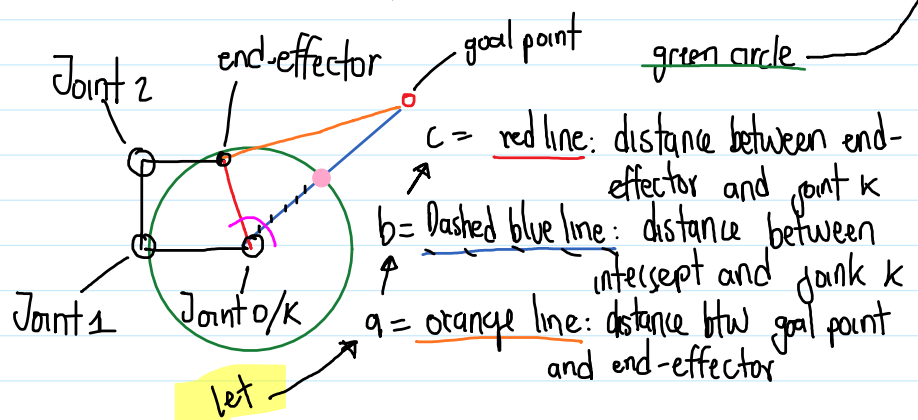
2. Using HTM from joint k to end-effector to get distance from joint k to end effector. (This distance represents the radius of the ~~circle~~ circle created when end effector is rotated about joint k). Save the coordinates of the end-effector -

3. find the intercept of the circle and the line
 Save this point

4. Find the distance between the intercept and point of joint k , and the distance between the intercept and the current position of the end-effector

- The idea is that you now have the lengths of a triangle we can get the angle to move joint k to get the end-effector to the line

i.e.



- The angle marked w/ a purple curve, is the angle we need to rotate joint k to, to get the end-effector to the line.

- \therefore this angle, A , can be calculated using the law of cosines / or the **THE DOT PRODUCT**

- ∴ This angle, A , can be calculated using the law of cosines or the

$$a^2 = b^2 + c^2 - 2bc \cos A$$

THE DOT PRODUCT

$$\cos A = \frac{a^2 - b^2 - c^2}{-2bc} \Rightarrow A = \cos^{-1} \left(\frac{a^2 - b^2 - c^2}{-2bc} \right)$$

5 Rotate point K by this angle. Then, if $K \neq 3$, increment K , else if $K = 3$, set K to 0. Return to step 1 with the new angles, and K value.

EQUATIONS

(x_E, y_E) - End-effector point/location.

D_{EJ} - Distance from end effector to joint K

(x_G, y_G) - Intercept point

D_{GJ} - Distance from goal point to point K

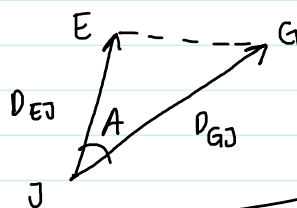
(x_J, y_J) - Joint K point.

E = End-effector
 G = Goal point
 J = Joint K

D_{EG} - Distance from end-effector to goal point

A = angle between vectors \vec{JE} and \vec{JG}

- The idea is to use these points to get the angle between two vectors



- This can be done w/ the law of cosines or the dot product. what I'll be using.

$$\vec{JE} \cdot \vec{JG} = \|\vec{JE}\| \|\vec{JG}\| \cos A$$

$$A = \cos^{-1} \left(\frac{\vec{JE} \cdot \vec{JG}}{\|\vec{JE}\| \|\vec{JG}\|} \right)$$

where: $\vec{JE} = \langle x_E - x_J, y_E - y_J \rangle$

$\vec{JG} = \langle x_G - x_J, y_G - y_J \rangle$

⊗ - You can perform the dot product in numpy using the `.dot()` function.