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 1 = 0
   " " 2 = 180
   " " 3 = 180

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).

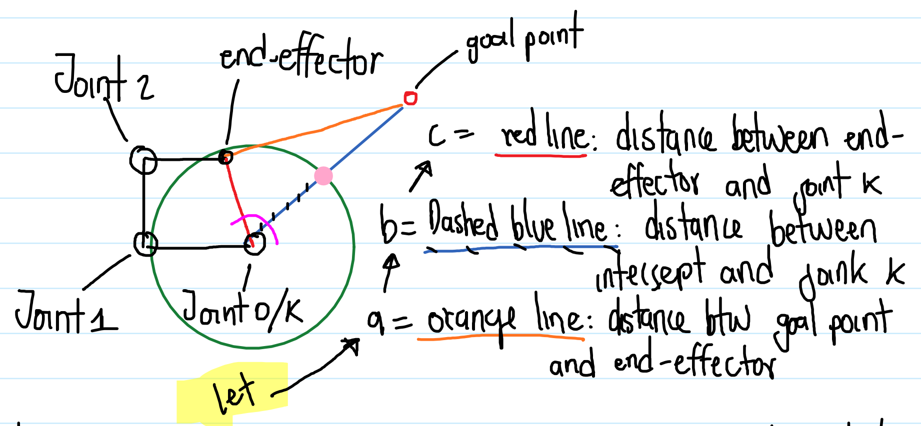
   2. Using HTM from joint k to end-effector to get distance from joint k to end effector. (This distance represents the radius of the circle created when end effector is rotated about joint k). Save the coordinates of the end-effector.

   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 side 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.

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

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

- So This angle, A, can be calculated using the law of cosines / or the $\boxed{\text{THE DOT PRODUCT}}$

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

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

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

<u>EQUATIONS</u>

(X_E, Y_E) - End-effector point / location.

(X_G, Y_G) - Intercept point

(X_J, Y_J) - Joint K point.

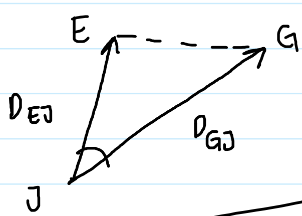
D_{EJ} - Distance from end effector to joint K

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

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

A = angle between vector $\vec{EJ}$ and $\vec{GJ}$

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



- This can be done w/ the law of cosines or $\boxed{\text{the dot product}}$ what i'll be using.

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

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

<u>where:</u> $\vec{JE} = \langle X_E - X_J, Y_E - Y_J \rangle$

$\vec{JG} = \langle X_G - X_J, Y_G - Y_J \rangle$

- You can perform the dot product in numpy using the $\boxed{.dot()}$ function.