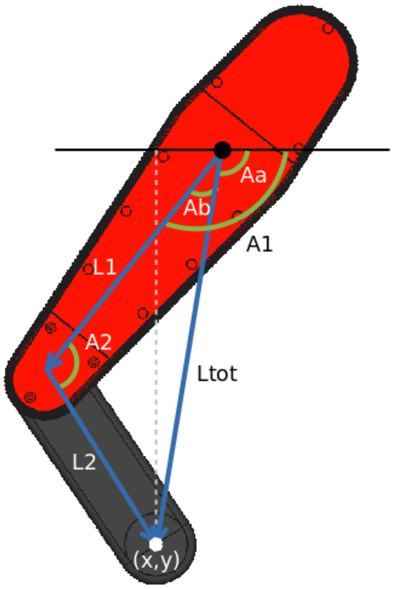
Alvaro Ferrán

Inverse Kinematics

This document's purpose is to explain how to find the arm's angles when given a position for the end-effector, calculating them geometrically.



In order to get to position (x,y) we need to find the angles A1 and A2 to which each motor has to move.
We will start by dividing A1 into two smaller angles: Aa and Ab. We calculate Aa using the Pythagorean Theorem:

$$Ltot = \sqrt{x^2 + y^2}$$

$$Aa = Acos(\frac{x}{Ltot})$$

In order to find Ab, we will use the Law of Cosines:

$$L2^2 = Ltot^2 + L1^2 - 2 \cdot Ltot \cdot L1 \cdot cos(Ab)$$

$$\rightarrow Ab = Acos(\frac{Ltot^2 + L1^2 - L2^2}{2 \cdot Ltot \cdot L1})$$

Thus,

$$A1 = Aa + Ab$$

To find A2 we simply follow the Law of Cosines once more:

$$Ltot^2 = L2^2 + L1^2 - 2 \cdot L2 \cdot L1 \cdot cos(A2)$$

$$\rightarrow A2 = Acos(\frac{L2^2 + L1^2 - Ltot^2}{2 \cdot L2 \cdot L1})$$

Finally, we implement this algorithm in Java for our Android app:

```
double[] inverseKinematics(double x, double y) {
   double[] a={0,0};

   double L1=300;
   double L2=200;

   double Ltot=Math.sqrt( Math.pow(x,2) + Math.pow(y,2) );
   double Aa= Math.acos(x/Ltot);
   double Ab= Math.acos( ( Math.pow(Ltot,2) + Math.pow(L1,2) -
       Math.pow(L2,2) ) / (2*Ltot*L1) );
   double A1=Aa+Ab;
   double A2= Math.acos((Math.pow(L2, 2) + Math.pow(L1, 2) -
       Math.pow(Ltot, 2)) / (2 * L2 * L1));

   a[0]=A1;
   a[1]=A2;

   return a;
}
```

Which may then be called:

```
double[] angles=inverseKinematics(x,y);
```