uStepper Robot Kinematics

uStepper Robot Arm Rev 4 Kinematics

by Mogens Groth Nicolaisen

Contents

uStepper Robot Arm Kinematics 1

To model the uStepper robot arm, first forward kinematic analysis is performed. The kinematic analysis is done analytical using basic trigonometry because of the Robot's simple structure. D-H[1] with reference changing matrix transformations from world coordinates to end effector could be done but is not required to derive kinematics for this simple setup.

1.1 Forward Kinematics

In the forward kinematics case, calculations are done to derive the end effector point from a given set of actuator angles (θ). **Figure 1.1** shows the basic geometry of the robot arm when looking from the side. Notice the offset from the end point to the "gripper" end point. This makes the model more genereic when using different actuator types. An offset from this point to the actuator is required though. Another feature to notice is that when changing θ_2 it is required to move both Primary and Secondary gear equally to maintain θ_3 - this has to be accounted for in the implementation phase.

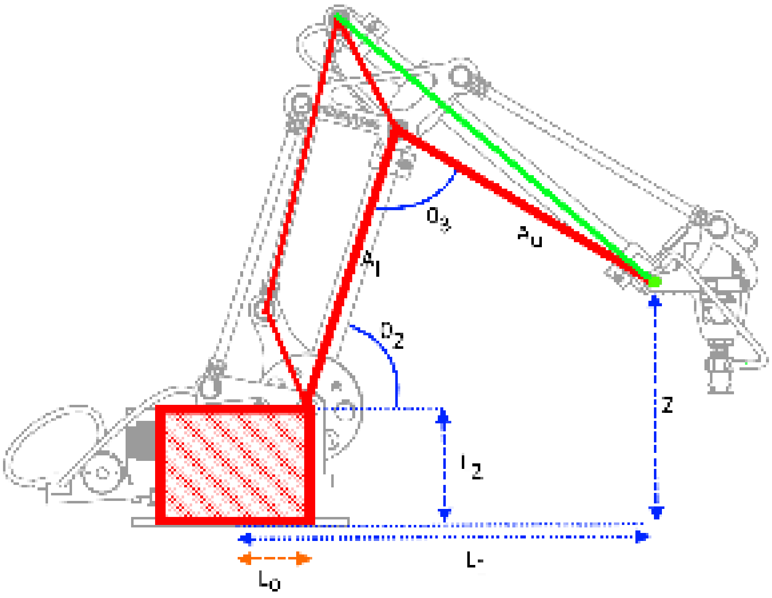

Figure 1.1: *Side view of the robot, with simple graphics applied to help deriving the kinematic equations.*

From **Figure 1.5** it is easily seen that to account for the offset from end-point to actuator end-point an offset in Z and L_1 is required. These are denoted Z_o and L_{1o} respectively and for the actuator on **Figure 1.5** Z_o would be negative while L_{1o} would be positive. The offsets are omitted in the following derivations to ease the readability. In the code they are added.

[1] Denavit–Hartenberg

Using the geometry presented in **Figure 1.1** Z is derived by adding right triangles resulting in the more detailed **Figure 1.2**

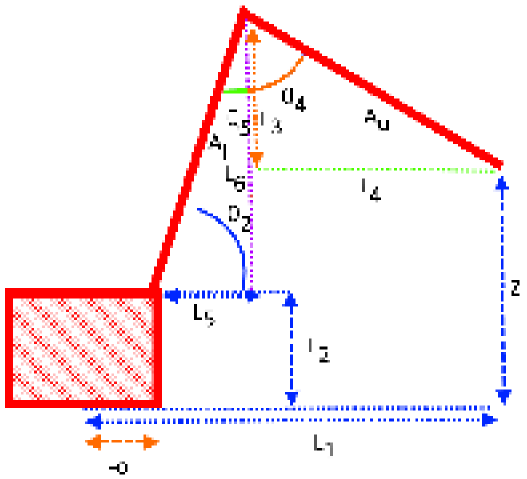

Figure 1.2: *Side view of the robot arm representation for kinematics calculations.*

Using **Figure 1.2** Z will be derived as shown **Equation (1.1)**:

$$Z = L_2 + L_6 - L_3 \qquad [mm] \quad (1.1)$$

Where L_3 and L_6 will be derived using simple trigonometry and L_2 is a design parameter from **Table 1.1**. First L_6 is derived in **Equation (1.3)**

$$sin(\theta_2) = \frac{L_6}{A_l} \qquad (1.2)$$

$$L_6 = sin(\theta_2)A_l \qquad [mm] \quad (1.3)$$

To derive L_3 it is required to know θ_4, which is possible to derive from θ_5 found in **Equation (1.4)**:

$$\theta_5 = 180^\circ - (90^\circ + \theta_2) \qquad [^\circ] \quad (1.4)$$

Where the 180° comes from the angle sum of a triangle, and the 90° from the known right angle. The angle θ_2 was given as input to the forward kinematics as discussed previously.
θ_4 is derived in **Equation (1.5)**:

$$\theta_4 = \theta_3 - \theta_5 \qquad [^\circ] \quad (1.5)$$

And L_3 is found from **Equation (1.7)**

$$cos(\theta_4) = \frac{L_3}{A_u} \qquad (1.6)$$

$$L_3 = cos(\theta_4)A_u \qquad [mm] \quad (1.7)$$

Z can now be derived using **Equation (1.1)** stated here again for convenience:

$$Z = L_2 + L_6 - L_3 \qquad \text{[mm]} \quad (1.8)$$

Now that Z is derived, X and Y is left. To derive those the view is changed to the top view in **Figure 1.3**.

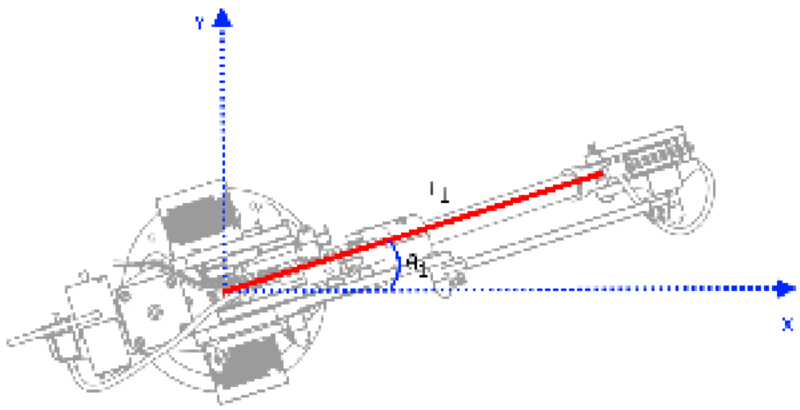

Figure 1.3: *Top view of the robot arm with world coordinates on top.*

As also shown on **Figure 1.1**, **Figure 1.3** shows the strecth L_1. Additionally the rotation angle θ_1 is marked on **Figure 1.3**. Using simple trigonometry X and Y is derived in **Equation (1.12)**.

$$cos(\theta_1) = \frac{X}{L_1} \qquad (1.9)$$

$$X = cos(\theta_1)L_1 \qquad \text{[mm]} \quad (1.10)$$

$$sin(\theta_1) = \frac{Y}{L_1} \qquad (1.11)$$

$$Y = sin(\theta_1)L_1 \qquad \text{[mm]} \quad (1.12)$$

Looking at **Figure 1.2** it is obvious that L_1 is the sum of L_4 and L_5. L_5 is derived using regular trigonometry in **Equation (1.14)**.

$$cos(\theta_2) = \frac{L_5}{A_l} \qquad (1.13)$$

$$L_5 = cos(\theta_2)A_l \qquad \text{[mm]} \quad (1.14)$$

And finally L_4 is found from **Equation (1.16)**.

$$sin(\theta_4) = \frac{L_4}{A_u} \qquad (1.15)$$

$$L_4 = sin(\theta_4)A_u \qquad \text{[mm]} \quad (1.16)$$

Arriving at L_1 in **Equation (1.17)**.

$$L_1 = L_4 + L_5 + L_a \qquad \text{[mm]} \quad (1.17)$$

Where L_o is the constant offset from rotation center.
To sum up, Z, X and Y is found by the following combined equations (from the previous derivations):

$$Z = L_2 + L_6 - L_3 \quad [mm] \quad (1.18)$$

$$Z = L_2 + sin(\theta_2)A_l - cos(\theta_3 - (90^\circ - \theta_2))A_u \quad [mm] \quad (1.19)$$

$$X = cos(\theta_1)L_1 \quad [mm] \quad (1.20)$$

$$X = cos(\theta_1)(sin(\theta_3 - (90^\circ - \theta_2))A_u + cos(\theta_2)A_l + L_o) \quad [mm] \quad (1.21)$$

$$Y = sin(\theta_1)L_1 \quad [mm] \quad (1.22)$$

$$Y = sin(\theta_1)(sin(\theta_3 - (90^\circ - \theta_2))A_u + cos(\theta_2)A_l + L_o) \quad [mm] \quad (1.23)$$

1.1.1 Summing Up

To implement the forward kinematics the former derived equations can be arranged to reduce computations required. This will be done in the following, and a code example will be presented.
The input for this is:

- θ_1 - rotation angle
- θ_2 - shoulder angle
- θ_3 - secondary gear angle

And the required equations are:

$$Z = L_2 + sin(\theta_2)A_l - cos(\theta_3 - (90^\circ - \theta_2))A_u \quad [mm] \quad (1.24)$$

$$k_1 = (sin(\theta_3 - (90^\circ - \theta_2))A_u + cos(\theta_2)A_l + L_o) \quad [\,] \quad (1.25)$$

$$X = cos(\theta_1)k_1 \quad [mm] \quad (1.26)$$

$$Y = sin(\theta_1)k_1 \quad [mm] \quad (1.27)$$

A code example snippet is provided in the following (note that Z_o and L_{1o} are added here as discussed in **Section 1.1**):

```
void uStepperRobotics::FWKinematic(float& x, float& y, float& z, float theta1, float theta2,
    float theta3)
{
  // From the documentation the elbow angle theta3 is the manipulated through the secondary
    gear
  // The primary gear is manipulating the shoulder angle theta2

//REMEMBER TO ADD OFFSET TO ACTUATOR!
  z = L2 + sin(theta2)*Al - cos(theta3 - (90 - theta2))*Au + Zo; //offset in the Z directon
    for the actuator is added here

  k1 = sin(theta3 - (90 - theta2))*Au + cos(theta2) Al + Lo + L1o;//offset in the L1 directon
     for the actuator is added here

  x = cos(theta1)*k1;

  y = sin(theta1)*k1;

}
```

1.2 Inverse Kinematics

To control the robot end-effector reference positions are given in the world coordinate system, which then has to be translated to a given set of actuator angles achieving these reference positions. I.e. in the inverse kinematics case, the desired end-effector coordinates are known and from these the actuator angles must be derived.
The starting point of the inverse kinematics derivation is the rotation angle. Looking at **Figure 1.4** it is obvious that simple trigonometry where a right angle triangle is constructed from the end of L_1 down to the x-axis, can be used.

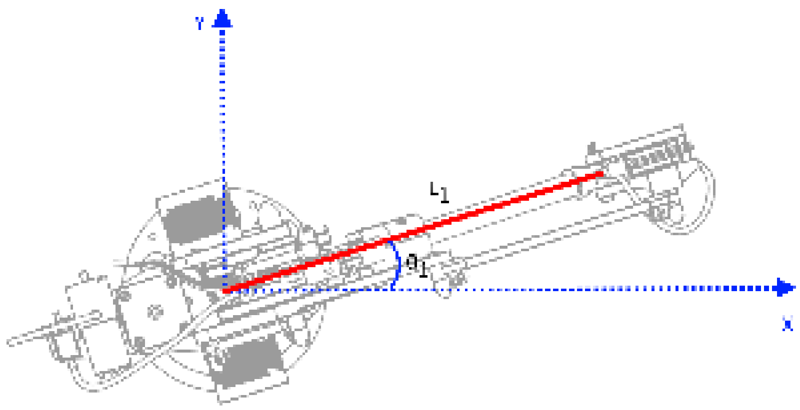

Figure 1.4: *Top view of the robot arm with world coordinates on top.*

To derive θ_1 **Equation (1.28)** is used.

$$\theta_1 = atan2(Y, X) \quad [°] \quad (1.28)$$

Using $atan2$ provides solutions for $X = 0$ and $X < 0$ as opposed to the regular arcus tangens function where solutions for these cases has to be handled separately.
For the next calculations the length of L_1 is required and therefore found in **Equation (1.29)**.

$$L_1 = \sqrt{X^2 + Y^2} \quad [mm] \quad (1.29)$$

To derive the remaining two angles a new figure is constructed (**Figure 1.5**).

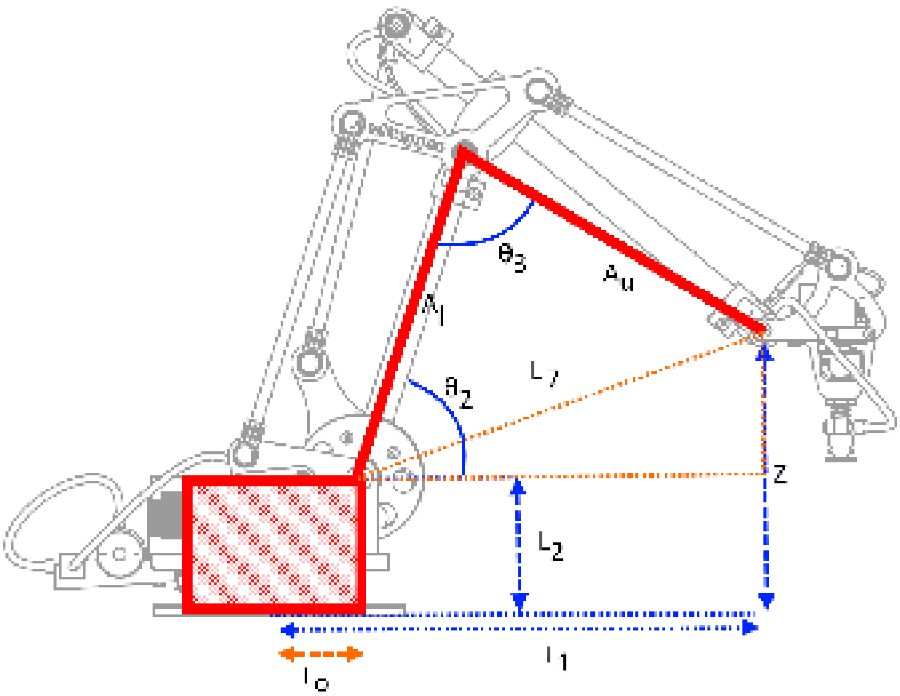

Figure 1.5: *Side view of the robot, with simple graphics applied to help deriving the kinematic equations.*

First θ_3 is derived by use of the cosine rule which requires knowledge of L_7 derived in **Equation (1.30)**.

$$L_7 = \sqrt{(L_1 - L_o)^2 + (Z - L_2)^2} \quad [mm] \quad (1.30)$$

For the next calculations a couple of solutions for trigonometric problems have to be considered:

$$cos(\theta) = x \rightarrow \theta = atan2\left(\pm\sqrt{1-x^2}, x\right) \quad (1.31)$$

$$sin(\theta) = x \rightarrow \theta = atan2\left(x, \pm\sqrt{1-x^2}\right) \quad (1.32)$$

As well as the cosine rule:

$$cos(A) = \frac{b^2 + c^2 - a^2}{2bc} \quad (1.33)$$

$$(1.34)$$

And θ_3 can be found in **Equation (1.36)**.

$$cos(\theta_3) = \left(\frac{A_l^2 + A_u^2 - L_7^2}{2A_l A_u}\right) \quad (1.35)$$

$$\theta_3 = atan2\left(\pm\sqrt{1 - \left(\frac{A_l^2 + A_u^2 - L_7^2}{2A_l A_u}\right)^2}, \left(\frac{A_l^2 + A_u^2 - L_7^2}{2A_l A_u}\right)\right) \quad [°] \quad (1.36)$$

θ_2 can be found by a combination of the orange right angle triangle in **Figure 1.5** and the triangle above that once again.

$$\theta_2 = atan2\left(\left(\frac{(Z-L_2)}{L_7}\right), \pm\sqrt{1-\left(\frac{(Z-L_2)}{L_7}\right)^2}\right) \tag{1.37}$$

$$+\, atan2\left(\pm\sqrt{1-\left(\frac{L_7^2+A_l^2-A_u^2}{2L_7A_l}\right)^2}, \left(\frac{L_7^2+A_l^2-A_u^2}{2L_7A_l}\right)\right) \quad [^\circ] \tag{1.38}$$

1.2.1 Summing Up

With implementation in mind a code example for the implementation is suggested. The inverse kinematics has X, Y and Z as input (note that Z_o and L_{1o} are added here as discussed in **Section 1.1**). To simplify matters the constants/offsets are subtracted as early as possible which is seen in the code snippet.

```
void uStepperRobotics::InvKinematic(float& theta1, float& theta2, float& theta3, float x,
    float y, float z)
{
  x = x - cos(theta1)*L1o;
  y = y - sin(theta1)*L1o;
  z = z - Zo - L2;

  theta1 = atan2(y,x);//rotation is denoted as theta1 in the documentation.

  float L1 = sqrt(x*x + y*y) - Lo;

  float L7 = sqrt(L1*L1 + z*z);

  float a = z/L7;
  float b = (L7*L7 + Al*Al - Au*Au)/(2*L7*Al);
  float c = (Al*Al + Au*Au - L7*L7)/(2*Al*Au);

  theta2 = (atan2(a,sqrt(1 - a*a)) + atan2(sqrt(1 - b*b),b));

  theta3 = atan2(sqrt(1 - c*c),c);

  theta1 = theta1* 180 / 3.1415;
  theta2 = theta2* 180 / 3.1415;
  theta3 = theta3* 180 / 3.1415;

}
```

1.3 Implementation

In this section the implementation will be discussed. This includes looking at the physical system - the uStepper Robot Arm Rev 4, and deriving the constants neede for the forward and inverse kinematics.

1.3.1 Pre-defined Constants

A table of constants is generated for use in the calculations (**Table 1.1**).

Variable	Value	Description
A_l	177 mm	Lower arm length
A_u	180 mm	Upper arm length
L_2	71 mm	Height from base to gears horizontal axis
L_o	41 mm	Offset in the L_1 direction
L_{1o}	40 mm	Actuator offset in L_1 direction
Z_o	−77 mm	Actuator offset in Z direction

Table 1.1: *Constants required for kinematic calculations.*

The constants are found on the robot arm as shown in **Figure 1.6**.

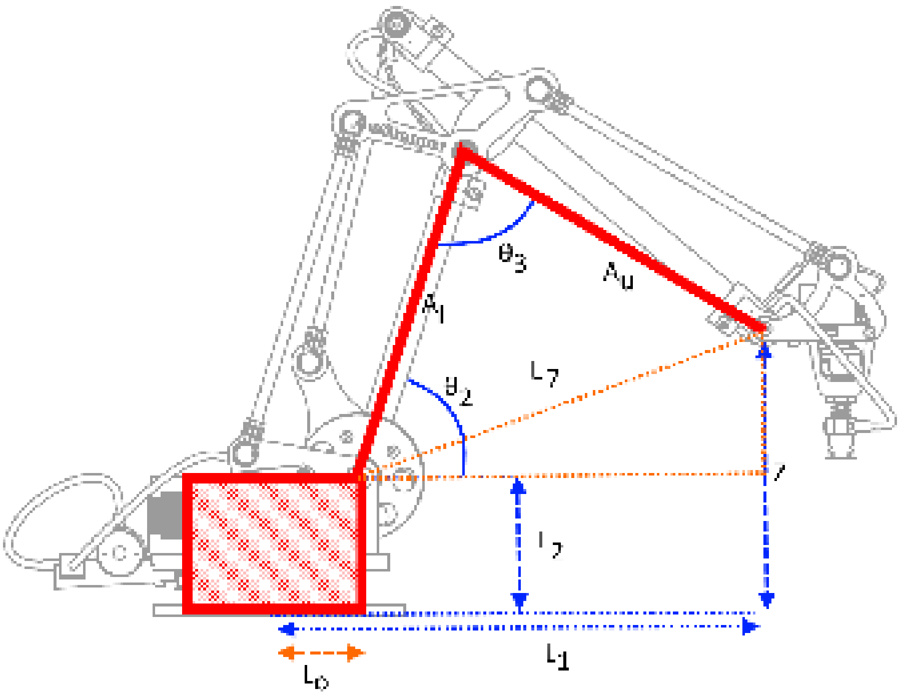

Figure 1.6: *Side view of the robot, with simple graphics applied to show the constants use in the calculations.*

The constants in **Table 1.1** is derived by measuring the robot as shown on **Figure 1.6**. Furthermore the measurements L_{r1}, L_{r2}, L_{r3} and L_{r4} are shown here to emphasize that the pauirwise have to be of equal length, e.g. the ones marked L_{r1} is required to be of equal lenght as well as the two L_{r2} etc. This is required to get the benefits of having two parallelograms as part of the robots mechanical structure.

1.3.2 Gear Ratios

All calculated angles from the kinematics are subject to a gear ratio calculation to e.g. get the inverse kinematic angles to motor angles. The gear ratio is the same for all three axis on the uStepper Robot Arm Rev 4 - 5.1 : 1. As an example, the calculated θ_1 from the inverse kinematics would be multiplied by 5.1 before it is sent on to the motor.

1.3.3 Physical Limitations

It is quite important to implement functionality that includes the robots physical limitations in the calculations. E.g. a world coordinate of $(1, 1, 0)$mm is not possible to reach since that is some place in the robot frame. It is obvious from the previous discussions and **Figure 1.2** (shown again in **Figure 1.7**) that it is not possible for the arm to reach within L_o of 41 mm from the base - and more has to be added because of other physical limitations.

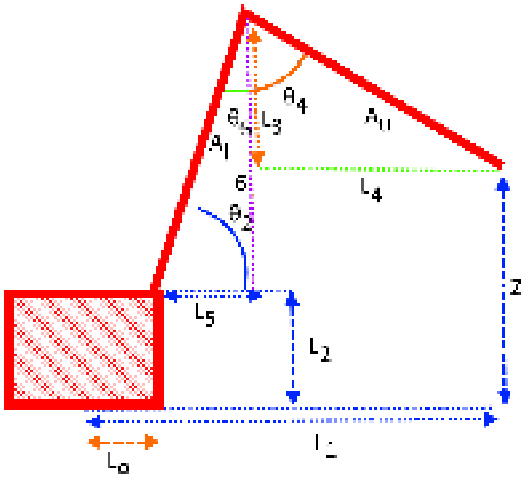

Figure 1.7: *Side view of the robot arm representation for kinematics calculations.*

Limitations will be handled in joint space and thus the following limitations to angles must be considered:

- Limit1: The gear angles can never go negative.
- Limit2: The secondary gear must be kept behind the main gear by 2° to avoid collision between the gears.
- Limit3: From homing the rotating axis θ_1 it is necessary to limit it to going between 0° and 358° because of the mechanical stop.
- Limit4: The main gear (shoulder) must never go below 10° from horizontal (i.e. θ_2 must always be larger than 10°) to avoid skewing the parallelograms of the robot.