Delta Bot Inverse Kinematics

Reference: Melgar, E. and Diez, C. (2012). Arduino and Kinect Projects, Chapter 13.

Introduction

The objective is to calculate the servo angle for each leg of the delta bot, given a required position in X, Y, and Z coordinates.

Leg Segments and Joints

The Thigh is connected to the Base at the Hip, as shown in Figure 1A. The Hip connection can be considered to be a "pinned" simple joint. Thus the Thigh always moves in a single plane. However, the Knee and Ankle joints can be modeled as ball joints, allowing the Ankle to pivot outward in the +Z and –Z directions. The Hip connection for each leg is modeled as being at a constant distance from the center of the base plate – a distance referred to in the equations as *Base*. Similarly, all three legs terminate at an Ankle joint connected to an effector plate that holds the working tool of the Delta Bot. All three ankle connections are modeled as being at a constant distance from the center of the effector plate – a distance referred to in the equations as *Effector*.

Inverse Kinematics in Two Dimensions

Let's first consider movement just in the 2-D plane that can be swept by the Thigh rotating about the Hip. The plane will be modeled in X-Y coordinates, with the origin at the center of the Base. The Effector (i.e. the gripper, work tool, etc.) is some fixed distance from the Ankle, usually the radius of the hub connecting the Ankles of the various legs. The Effector (the free end of the "Effector" line in Figure 1a) is at the user-specified location (x,y).

The length of the line segment c in Figure 1a can be determined[1] from the given (x,y) using the equation

$$c = \sqrt{[(x + Effector) - Base]^2 + y^2}$$

Modified from the Text

Then, using the Law of Cosines:

$$\alpha = cos^{-1}\left[\frac{(-[shin^2] + thigh^2 + c^2)}{2 * thigh * c}\right]$$

Modified from the Text

Next, determine the angle "beta"[2] and *servoAngle*:

$$\beta = atan2[y, (x + Effector - Base)]$$

Modified from the Text

$$servoAngle = \ \alpha - |\beta|$$

Modified from the Text

The angle α is always positive. If α is larger than the absolute value of β, then the thigh strut is angled up above the horizontal and *servoAngle* is positive. If α is smaller than the absolute value of β, then the thigh strut is angled below the horizontal and *servoAngle* is negative.

[1] The equation for the distance between two points in the XY plane, (x_1,y_1) and (x_2,y_2), is $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

[2] Here we are using the function "atan2(y,x)" – closely related to $\tan^{-1}\left(\frac{y}{x}\right)$ but with a range of return values being (-π,π]. For more information on atan2() see https://en.wikipedia.org/wiki/Atan2.

Inverse Kinematics in Three Dimensions

Now let's assume that the Ankle is shifted out of the X-Y plane, giving the Effector a location of (x,y,z) as shown in Figure 1b. Note the unusual arrangement of axes – the Y axis is vertical and the Z axis comes out of the page toward the reader. The length of the line segment a in Figure 1b is then:

$a = \sqrt{shin^2 - z^2}$, and

$$\alpha = cos^{-1}\left[\frac{(-[a^2] + thigh^2 + c^2)}{2 * thigh * c}\right]$$

The equations for c, β, and servoAngle remain unchanged.

Church of the Universal Coordinate System

We can now calculate the required servo angle as a function of the Effector location (x,y,z) relative to a coordinate system defined by the Base and the sweep of the Thigh. But a problem arises – the typical Delta Bot has a least three legs (sometimes many more), and the equations were derived relative to the plane of a leg. Which leg should be used as the basis of the user coordinate system?

The solution is to specify the target Effector location in a universal coordinate system, independent of any particular robot leg. The specified coordinates (x',y',z') can then be translated into the coordinate system for any given leg by using standard coordinate rotational transformations.

The universal coordinate system will be laid out similar to the coordinate system shown in Figure 1b, with the X axis to the right, the Y axis in the vertical position, and the Z axis coming out of the page. Assume that the *i*th leg is rotated away from the X axis **counterclockwise** around the Y axis by an angle ϕ_i. The universal coordinates (x',y',z') can be converted into coordinates in the leg's coordinate system (x_i,y_i,z_i) using the following[3]:

$$x_i = x' * \cos(\phi_i) - z' * \sin(\phi_i)$$

$$z_i = x' * \sin(\phi_i) + z' * \cos(\phi_i)$$

$$y_i = y'$$

Note that if a leg is located on the X-axis in the universal coordinate system, then for that leg x_i = x', z_i = z', and y_i = y'. It is also possible to translate the coordinate systems if the base of the Delta Bot is not centered over the origin of the universal coordinate system, but that will be left as an exercise for the reader.

Once x_i, y_i, and z_i have been determined for the *i*th leg, the Inverse Kinematics in Three Dimensions can be used to calculate the servo angle for that leg. Rinse and repeat for all other legs, varying the angle ϕ_i.

[3] This sort of math is generally referred to as a "rotation matrix". For more information, see https://en.wikipedia.org/wiki/Rotation_matrix

Summary

Given a desired position at the coordinates (x', y', z') and a leg "i" rotated counterclockwise away from the X-axis by an angle of ϕ_i:

$$x_i = x' * \cos(\phi_i) - z' * \sin(\phi_i)$$

$$y_i = y'$$

$$z_i = x' * \sin(\phi_i) + z' * \cos(\phi_i)$$

$$c_i = \sqrt{(x_i + Effector - Base)^2 + y_i^2}$$

$$a_i = \sqrt{shin^2 - z_i^2}$$

$$\alpha_i = cos^{-1}\left[\frac{(-[a_i^2] + thigh^2 + c_i^2)}{2 * thigh * c_i}\right]$$

$$\beta_i = atan2[y_i, (x_i + Effector - Base)]$$

$$servoAngle_i = \alpha_i - |\beta_i|$$

Implementation Issues

The real world is never as simple as the mathematics can make it seem. Several issues can arise when implementing these equations to control a delta arm.

First, the servoAngle in the equations and in Figure 1a will have values ranging from -90 degrees to 90 degrees. However, the servo library used to control the servos frequently will use a range of angles from 0 degrees to 180 degrees. In this situation, adding 90 degrees to the servoAngle calculated previously will result in angles compatible with the servo library.

Second, the desired position at the coordinates (x', y', z') is the position of the *center of the effector platform*. If you have a tool connected to the effector platform (a gripper, a suction head, or even just a stick), you may be more interested in controlling the position of the tool. In such a case you would subtract the length of the tool from the y' value to get the desired vertical location of the effector platform used in the equations.

It is possible to specify a desired position at the coordinates (x', y', z') that the delta arm cannot reach, given the physical dimensions of the thigh, shin, effector, and base. An easy way to filter out these positions prior to moving the arm is to check the equations for a, α, and servoAngle for invalid arguments. In the equation for a, the equation under the square root must be positive for a valid set of coordinates. In the equation for α, the argument of the $\cos^{-1}$ function must be a value between -1 and 1 for a valid set of coordinates. Lastly, any calculated value for servoAngle should be between -90 and 90 for a valid set of coordinates (unless you have added 90 degrees as described previously – then the valid servoAngle would be between 0 and 180 degrees).

Finally, when constructing the servo arm it is possible that the horns on the servos are not perfectly aligned. In such a situation positioning the servo at 45° might result in the servo horn pointing to 48° or 41°. In such a situation you may want to add a "bias", or correction factor, to the angle before calling the servo library to move the servo.

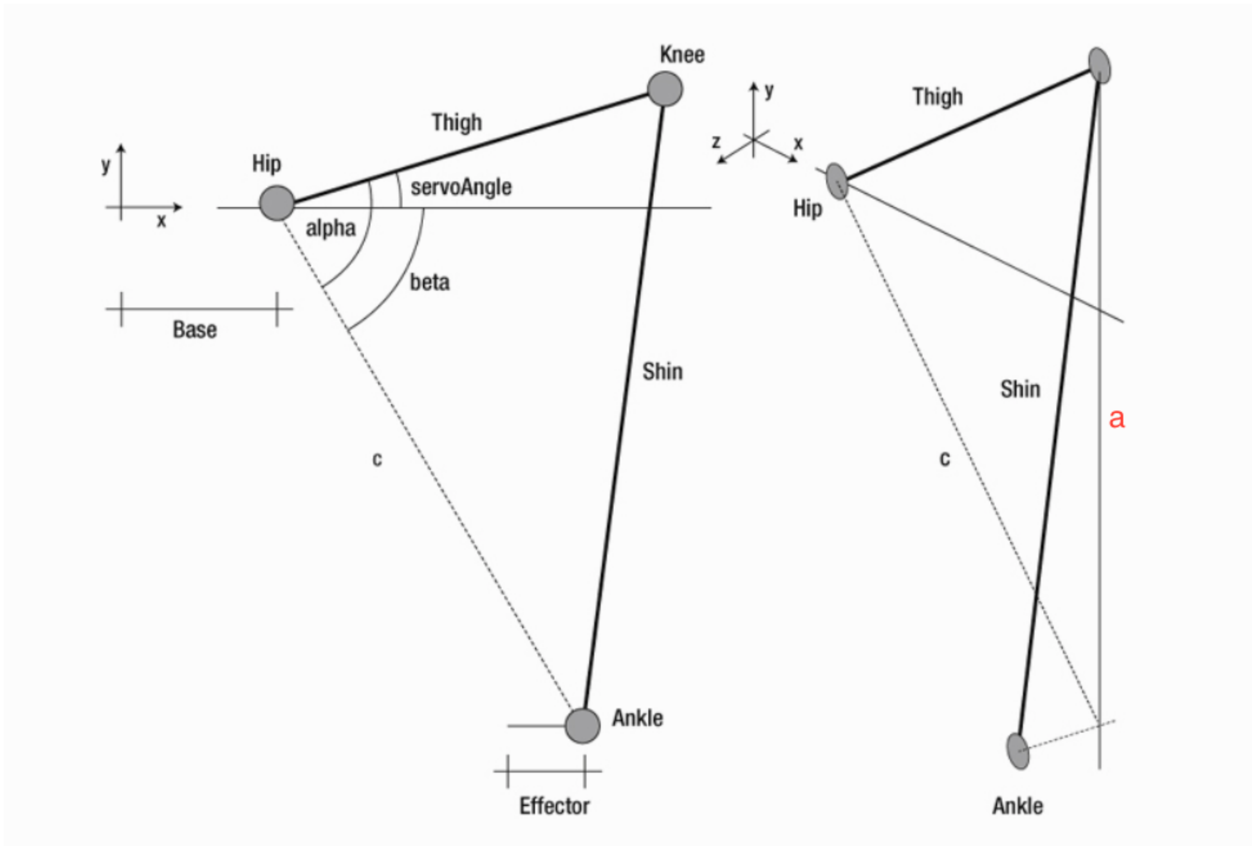

Figure 1a) Delta bot leg in two dimensions and Figure 1b) Delta bot leg with an ankle offset in the Z direction.