KITtyBot

A Small Quadruped Robot

KITtyBot is a small quadruped robot controlled by Arduino controller and equipped with 9 gram R/C micro servos. Unlike most robots this has a leg configuration in a more mammal like style rather than the more common insect layout. In the following pages the basic configuration of the robot is described in terms of theoretical background and hardware configuration.

The robot expects a flat and even surface and moves it's feet according to this. There is no feedback in the form of forces sensors or position feedback but the legs are moved as an open-loop control based on the Invers Kinematics derived in this report.

The report gives the theory to implement but there isn't any computer code described in detail here.

1 BACKGROUND

KITtyBot is inspired by several four legged robots like the various types designed by Boston Dynamics, IIT and ETH. The aim was to design a robot capable of walking and maneuvering on a flat surface with the use of hobby equipment like Arduino and normal R/C servos. When I asked for cute names for my robot many had cat or kitten inspired suggestions and in the end the name KITtyBot was chosen. It also includes the possibility to produce kits for the robot.

There are many legged robots, both as kits and home built, but a vast majority have a more insect like configuration, i.e. the legs in each corner points outwards. The mammal configuration is less common and is a little more challenging. It tips over more easily and has to have three legs on the ground forming a three pod with the center of gravity within it's support points.

The robot is designed to work on a flat end level surface. There is no sensing in the feet or feedback from joints, the joints are moved in an open-loop control and puts the feet where the ground is expected to be. But in order to know how to move the joints the Kinematics and Inverse Kinematics need to be worked out. The aim with this report is to describe how this was done.

2 CONFIGURATION

The basic configuration of the robot is shown in Figure 1. It shows a simplified 3D image of the robot with body, joints and leg parts. As seen the joint connecting to the body is a hip joint allowing movements in two degrees of freedom. The joint connecting the upper and lower leg parts is a knee joint with one degree of freedom.

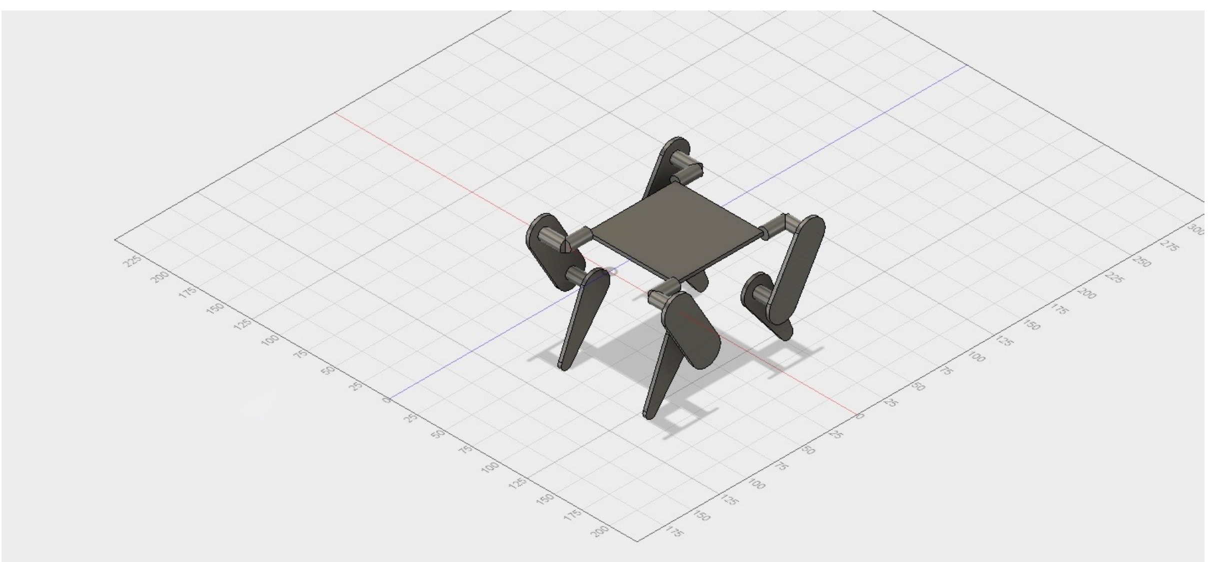

Figure 1, Basic Configuration

3 KINEMATICS

In order to define the kinematics of the robot the coordinate system is defined with x, y and z axis. The x axis points in the walking direction (+x forward and -x reverse). The y axis in the side direction, (+y to the left and -y to the right). The z axis is the vertical direction (+z up and -z down).

When defining a leg the subscripts "front", "rear", "left" and "right" is used, as an example x_{FL}. There is a risk of confusion between "right" and "rear". In this cases the subscripts uses two letters, i.e. x_{Re} or x_{Ri}. Subscripts are used only to the extent needed for understanding. General equations often don't have them and if equations are generalized to one pair of legs (left, right, front or rear) only this subscript is used.

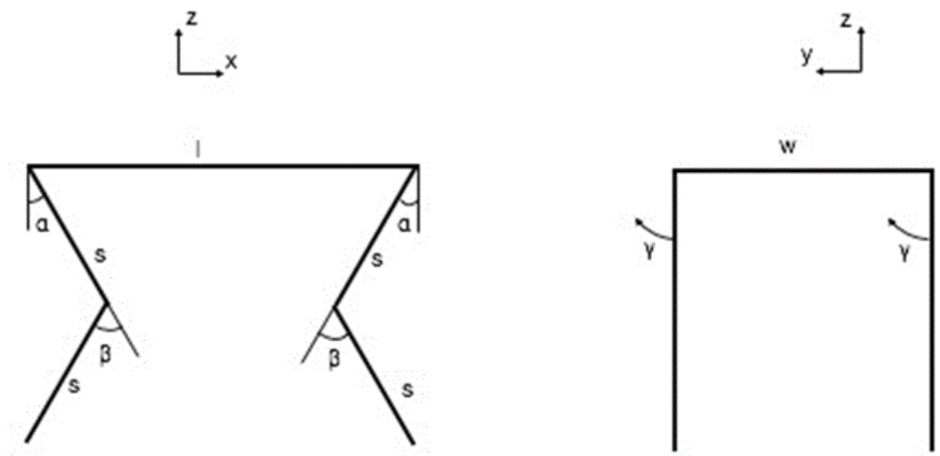

Figure 2, Robot seen from side (facing right) and from rear

Figure 2 shows the different angles and dimensions needed to define the robots foot positions. The position of the foot relative to its attachment to the body can be expressed by a series of trigonometric equations.

For the front legs we have the following equations:

$$x_F = -s(\sin\alpha + \sin(\alpha - \beta)) \qquad 1$$

$$y_F = s(\cos\alpha + \cos(\alpha - \beta))\sin\gamma \qquad 2$$

$$z_F = -s(\cos\alpha + \cos(\alpha - \beta))\cos\gamma \qquad 3$$

For the rear legs the set of equations become:

$$x_{Re} = s(\sin\alpha + \sin(\alpha - \beta)) \qquad 4$$

$$y_{Re} = s(\cos\alpha + \cos(\alpha - \beta))\sin\gamma \qquad 5$$

$$z_{Re} = -s(\cos\alpha + \cos(\alpha - \beta))\cos\gamma \qquad 6$$

The parameters for the robot are defined:

Length of each leg part, s = 50 mm

Length of robot (distance between hip joints in x directions).

l = 120 mm

Width of robot (y wise distance between hip joints).

w = 80 mm

The robot has a starting configuration, i.e. angles and positions that is the starting point for all movements and gaits. It was decided early on that slightly bent knees with the feet being positioned directly under the hip joints should be the starting position. The chosen angles were.

α_0 = 30°

β_0 = 60°

These are angles were chosen due to their ease to work with (easy to draw with elementary tools and cos and sin for 30° and 60° are known to most people without the need for an advanced calculator). Beside from this these angles give good maneuverability combined with a rather “erect” appearance.

The intention was to have the γ angel to zero but it was discovered early that sideways movements of the body were required so it wouldn’t tip over. In order to further increase stability a small toe out of 10 mm was chosen. By using this distance, α and β angles a value γ_0 can be calculated:

$$\gamma_0 = sin^{-1}\frac{to}{s(cos\,\alpha + cos(\alpha - \beta))} \qquad 7$$

With the value of toe out:

to = 10 mm

We get:

γ_{0L} = 6,63°

This is on the left side legs. For the right side the toe out works in negative y direction and we get a value of:

γ_{0Ri} = -6,63°

One last important parameter is calculating the robots height above ground, i.e. the distance from hip joint to foot in z direction. By simply using equations 3 or 6 above and the values for s, α_0, β_0, γ_0 we get:

h_0 = 86,0 mm

Note: The z value according to Equations 3 and 6 is in fact negative h_0 (the origin is at hip joint and the z axis point upwards). Here we only want to calculate the vertical distance between the hip and foot without any sign showing direction.

The height points only in z direction and the toe out in y direction. A leg length can be defined by using the Pythagorean Theorem:

$leg_0 = \sqrt{h_0^2 + to^2}$ = 86,6 mm

Note: This happens to be corresponding with the term s(cosα + cos(α-β)) in equations 2, 3, 5 and 6 above if α_0 and β_0 are used. And anyone who knows trigonometry, and has learned the solutions for 30° and 60°by heart, sees that this is the solution, $\sqrt{3/4}$, for cos 30° multiplied with 100 mm (2s).

4 INVERSE KINEMATICS

The equations above describe how the foot positions can be calculated if the joint angles are known. A starting configuration was also defined. Angular movements of the joints based on desired feet movements must be calculated. This is called inverse kinematics (IK). There are several methods to do IK on robots. Since this is a rather simple robot and has a well-defined starting position we can use a purely analytical solution based on inverse trigonometry.

This is done by first calculating a leg length caused by changes in foot positions (Δx, Δy, Δz). There is one equation for the left side legs and one for the right side:

$$leg_L = \sqrt{\Delta x^2 + (\Delta y + to)^2 + (h_0 - \Delta z)^2} \qquad 8$$

$$leg_{Ri} = \sqrt{\Delta x^2 + (\Delta y - to)^2 + (h_0 - \Delta z)^2} \qquad 9$$

The angle β is dependent only on this leg length.

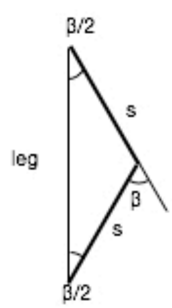

Figure 3, Leg length

The leg length is:

$$leg = s\cos\frac{\beta}{2} + s\cos\frac{\beta}{2} = 2s\cos\frac{\beta}{2}$$

From this β is calculated:

$$\beta = 2\cos^{-1}\frac{leg}{2s} \qquad (10)$$

The angle α is dependent of the displacement of the foot in x direction, see Figure 4. In case of a zero displacement the angle is half the value of β. In other cases a Δx movement adds (or subtract to this value)

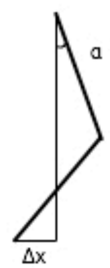

Figure 4, Alfa angle

Due to the definitions outlined in the previous section there are different equations for the front and rear sides:

$$\alpha_F = \frac{\beta}{2} - \sin^{-1}\frac{\Delta x}{leg} \qquad (11)$$

$$\alpha_{Re} = \frac{\beta}{2} + \sin^{-1}\frac{\Delta x}{leg} \qquad (12)$$

The last angle to calculate is γ. This is dependent on the previously calculated angles and the displacement Δy. Equations 2 and 5 can be used to calculate this value. There are different equations for left and right side:

$$\gamma_L = \sin^{-1}\frac{\Delta y + to}{s(\cos\alpha + \cos(\alpha - \beta))} \qquad (13)$$

$$\gamma_{Ri} = \sin^{-1}\frac{\Delta y - to}{s(\cos\alpha + \cos(\alpha - \beta))} \qquad (14)$$

This concludes the inverse kinematics of the robot. All desired foot movements, expressed as a "delta" from the starting configuration, can be calculated into angle changes. The servos do angular movements and are directly mounted on each joint so the angles proportionally corresponds to a servo output.

5 Using Inverse Kinematics, an Example

The inverse kinematics derived in the chapter above forms the code to move the legs on the robot. The input to the code is a requested foot movement expressed in x, y and z coordinates. With this new angles for each joint can be calculated and the R/C servos position themselves into that position. From this a new (x, y, z) input forms new joint angles and reposition the servos etc.

The best way to get a grip on how it works is to do an example:

We want the front right leg to be lifted 20 mm and moved 20 mm forward and 20 mm inwards (to the robot's centre line. Expressed in our coordinate system this would be:

Δx = 20 mm
Δy = 20 mm
Δz = 20 mm

Equation 9 gives:

$$leg_{FRi} = \sqrt{\Delta x_{FRi}^2 + (\Delta y_{FRi} - to)^2 + (h_0 - \Delta z_{FRi})^2} = 69{,}7 \text{ mm}$$

This is used in Equation 10 to calculate β:

$$\beta_{FRi} = 2\cos^{-1}\frac{leg_{FRi}}{2s} = 92°$$

Then the angle α is calculated (Equation 11):

$$\alpha_{FRi} = \frac{\beta_{FRi}}{2} - sin^{-1}\frac{\Delta x_{FRi}}{leg_{FRi}} = 29°$$

With this the γ angle finally can be calculated with Equation 13:

$$\gamma_{FRi} = sin^{-1}\frac{\Delta y_{FRi} - to}{s(\cos\alpha_{FRi} + cos(\alpha_{FRi} - \beta_{FRi}))} = 8{,}7°$$

But these angles into the original equations (1-3) to check:

x_{FRI} = 20 mm

y_{FRI} = 10 mm

z_{FRI} = -66 mm

This is a move of (20, 20, 20) from the original state of (0, -10, -86). Thus the inverse kinematics seem to work.

By using the start values for the three angles (30, 60, -6,63) angular movements of each joint can be derived:

$\Delta\alpha$ = -1°

$\Delta\beta$ = 32°

$\Delta\gamma$ = 15,3°

6 Exploring Walking Gaits

Once there is a working Inverse Kinematics function in place the robot can use it to move its feet into defined positions. Sequences for leg movement, commonly called walking gaits, have to be developed. One good description of walking gaits can be found on the Internet:

http://blog.oscarliang.net/quadruped-robot-gait-study/

What Oscar Liang calls the creep gait can be divided into sequences to be used in this robot. Apart from the movement of legs one at the time and the continuous moving of the body another element was added to improve stability. The body is shifted to the other side from the legs that are lifted. A principal sketch of how the gait works is shown in Figure 5. In theory the center of gravity is inside the tripod created by the remaining feet when one "rearmost" foot is lifted also when the body is centered. This was still not enough to assure stability so the sideways shifting was added. Another solution is to have the body even more positioned forward, a gait that might be studied in the future.

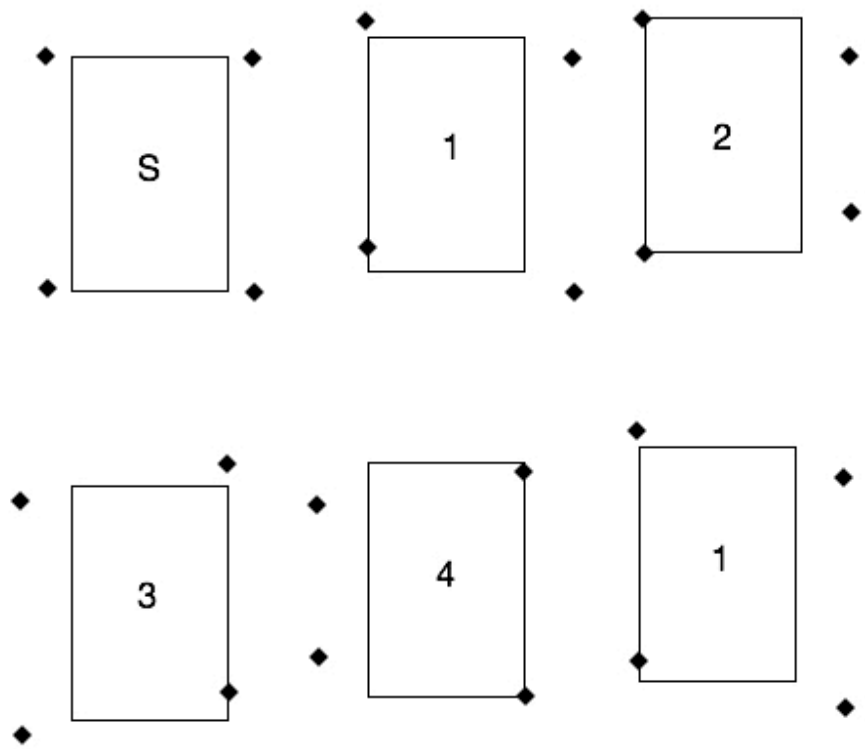

Figure 5, The walking gait

The sequences numbered 1 to 4 show how the gait is performed. At sequence 1 the left side legs are slightly in front of their starting position and the right slightly behind (actually the body has shifted forward to be between the left and right side feet). The body is shifted to the left to allow for lifting of the right side legs. In sequence 2 the rear right foot has moved one step length forward and the body is shifted forward one quarter of a step length, i.e. the feet in contact with the ground are moved backwards the distance while the rear right foot is lifted.

In sequence 3 the front right foot has also moved one step length forward, the body is moved ¼ step length and shifted to the right to allow for the left feet to be lifted. In sequence 4 the rear left foot is moved one step length and the body shifted forward ¼ step length. Finally we are back at sequence 1 as the front left foot is moved one step length at the same time as the body is shifted ¼ step length forward and to the left to allow for lifting of the right side feet. When returning to sequence 1 the body has moved one step length over the ground.

Each of these sequences consist of two parts. First one leg is lifted and in the second part the body is shifted by moving the three feet in contact with ground. At the same time the lifted foot is moved forward and put down. The lifted foot is placed on the ground one step length in front of its previous position. Since the body is moved ¼ step length at the same time the movement of this foot relative to the body is only ¾ step length.

Figure 5 does not show how to get from the starting position to sequence 1 and also not how to leave the gait and go back to the starting position. There is code for this, basically the body is shifted to the right, rear left and the front left feet are move one half step length and finally the body is shifted forward and to the left to assume the position at sequence 1.

The way of leaving is to go from sequence 1, move rear right and front right moved ½ step length and then center the body. The program checks at the end of a cycle, when going from sequence 4 back to 1, if there is still a forward command and then either continue with one more cycle or leave it and center.

In the first useable Arduino program the robot is controlled by push buttons and a standard step length of 40 mm was used. The robot maintains the height over ground of 86 mm during the complete cycle. A theoretical step length of at least 90 mm should be possible. These “leaps” caused other problems (robot tipping) but a code with analog input (joystick) with the possibility to vary the step length between 0 and 60 mm has been tested successfully.

7 MAKING A BODY TURN

Now when there is working gait for forward walking a way of turning the robot must be developed. A standing turn (twist) is shown in Figure 6. This can be obtained if all four legs are moved simultaneously into new positions.

In order to calculate this a new angle is introduced, ϕ. The body is twisted an angle $\Delta\phi$ from the starting position. This angular movement should be expressed in movements in x and y directions in order to make use of the IK functions.

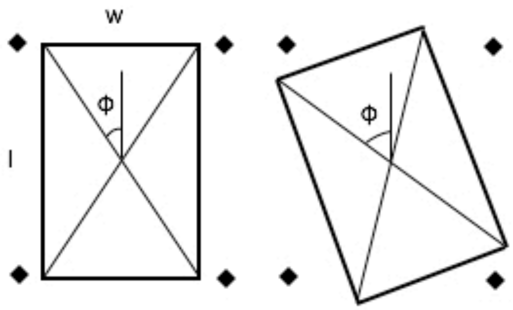

Figure 6, Body twist

If the origin is in the body centre the front left corner is at (l/2, w/2). This can also be expressed in the form:

$$x_{FL0} = r\,cos\,\varphi_{0FL}$$

$$y_{FL0} = r \sin \varphi_{0FL}$$

Where:

$$r = \frac{\sqrt{l^2 + w^2}}{2}$$

$$\varphi_{0FL} = \tan^{-1}\frac{w}{l}$$

If the body is twisted a new position of the corner becomes:

$$\Delta x_{FL} = r\,cos(\varphi_0 + \Delta\varphi) - x_{FL0} \qquad 15$$

$$\Delta y_{FL} = r\,sin(\varphi_{0FL} - \Delta\varphi) - y_{FL0} \qquad 16$$

It is quite straight forward to calculate the corresponding movements for the other three corners, simply add 90°, 180° and 270° (π/2, π and 3π/2 when radians are used) to ϕ_{FL0}.

Note: These equation describes how the corners of the body moves in order to obtain a twist. The feet (diamonds in the figure) have a toe out as described earlier. This must be added to the width in the equations above. Also if a twist is done with the feet in offset positions r and ϕ_0 values must be calculated for these.

A twist function was implemented in so that a body twist was first performed by simultaneous moving of four feet with Δx and Δy calculated according to the equations above. After this legs are lifted one by one and moved into their starting positions again. The body needs to be shifted sideways in order not to tip, after a turn the center of gravity remains, i.e. it is still on the edge of a tripod created by the three remaining feet.

8 HARDWARE SETUP

The aim with this report was to describe the theory used in the Arduino code for the robot. There is no code pasted into this report, since the code is constantly developing to incorporate new functions. At the time of writing the robot can perform straight walking forward and backwards, using the gait described above, and do turning (twisting) of the body.

On the robot there is a servo sequencer, SSC-32U. This can operate up to 32 R/C servos. The sequencer is powered by a 6V/1800 mAh NiMh battery carried under the “belly” of the robot. The SSC-32U receives serial commands from an Arduino that acts like a remote control. Currently there are three wires – Tx, Rx and GND – connecting the servo sequencer and the Arduino. The SSC has an xBee connection. This has been used to replace the three wires with a Bluetooth connection.

The SSC has a neat function that is used frequently on this robot. It can receive simultaneous servo commands with a common travel time. The travel speed on each servo is controlled so that they all leave their starting position and arrive at the finishing position simultaneously. This comes very handy when simultaneous commands like the twisting described above are done.

The Arduino controller has a joystick shield. In short a command issued generates a step length or turning angle and then functions for walking or turning divides it into sequences and order servo movements. Both push buttons and analog inputs from the joysticks have been tested.

9 FUTURE WORK

The aim with this robot was to design a mammal like robot, sort out the theory and develop working gaits to have it move on a flat surface wherever the person controlling it wants. By developing a walking gait and possibility for turning this has been achieved to a satisfactory level. Some future improvements can however be foreseen.

The SSC is a powerful but rather expensive control card for the servos. A future development could be to replace it by a small Arduino (or compatible) board and a servo control card. In such case some of the features of the SSC (like the simultaneous moving described above) must be programmed into that Arduino board. These two cards would also require less space on the robot itself.

Further a wireless communication between the remote and the robot could be developed. Bluetooth connection between the remote and the robot has been tested. If the SSC is replaced with Arduino we could try out a solution with other transceivers such as nRF24L01+. This is a cheaper solution but will need some more programming also on the robot.

Other future improvement could include exploring better walking and turning gaits. A trotting gait, lifting two feet at the time, was tested but wasn’t successful. The aim is to lift two diagonal and put them down in front while the two with contact to the ground shift the body forward. This is a more dynamic gait in such that the body only balances on two feet at the time but with fast enough movements this could be overcome. This didn't work out very well.

The aim with robotics is to have autonomous systems that walk and make decisions based on onboard sensors. This has not been fully explored here as the aim was more to explore the feasibility of the robot design as such as a walking robot. Future improvements could include on board sensors to avoid obstacles. There are Arduino controlled robots that can do this but maybe more computer power is required. Having a microcomputer, Raspberry Pi or similar, could be a solution to this.

Sensing feet is another field to explore. It could be done by force sensors under the feet or a small spring travel on the last joint connected to a position sensor. By this the robot can feel the ground and thus climb over small obstacles without tipping. This could be combined with gyros and accelerometers. There are components available for reasonable prices that can be tested.

There is plenty of room for improvements. Currently the robot can move around without tipping and that has been the goal for this project. It is suitable platform that can be further developed.