A Study and Kinematics Analysis of a Quadruped Robot by the Protype Building Leg

Abid Shahriar, Department of EEE, AIUB, Dhaka -1229, Bangladesh
Email: abidshahriar97@gmail.com

Abstract–Kinematics Analysis is a crucial part for multiple joint enabled Robot. For Mathematically moving a multi joint enabled robot, it needs some mathematical calculations to be done that way so the end effector's position can be determined with respect to the other connective joints involved and their respective frames in a specific co-ordinate system. For a locomotive quadruped robot, it is essential to determine two types of Kinematics for the Robot's leg position on the co-ordinate. For the part of Forward Kinematics, it measures the position and we can calculate the joint angles by using inverse kinematics. In this study, we mathematically analyse and derived the forward and the inverse kinematics of the quadruped robot and first we have done the simulation with Jupyter notebook in Python Environment for the mathematical analysis and verification and also test our Kinematics code on a protype build leg.

Keywords – Forward Kinematics, Inverse Kinematics, Walking Pattern, locomotion, transformation matrix, End effector, Co-ordinate system.

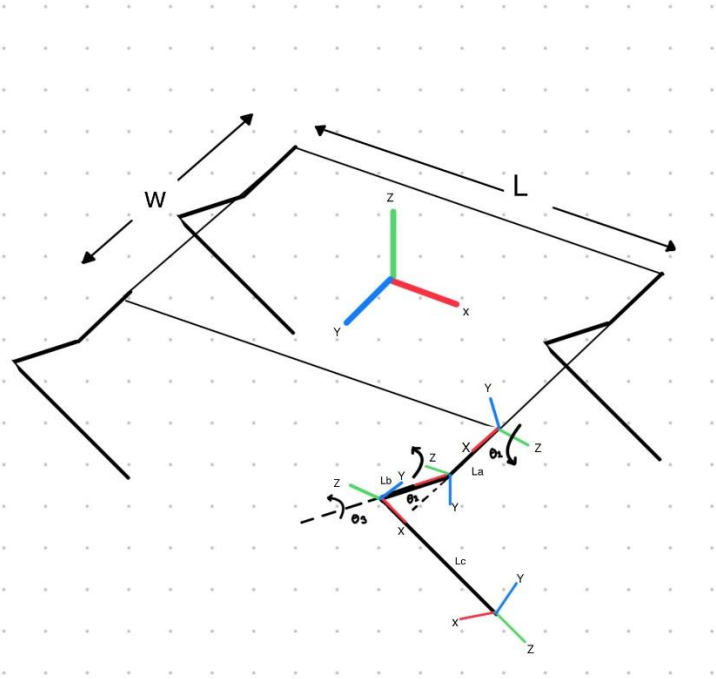
Introduction:

Bio-Inspired Robots are getting more popularity due to the capability of its ability maneuver through uneven surfaces and less energy conversion is needed over wheel based robots. The Quadrupped robot or the four legged robots are Bio-Inspired robots which is getting more concentration for its locomotion control. For the robot's leg to move mathematically and in a cotrolled and precise way Kinematics Analysis is a crucial part and these calculations can be done through Forward Kinematics and Inverse Kinematics study. Using Translational Matrix to calculate frame by frame calculation is one of the way of this approach [1]. Recently, there has been a significant focus on incorporating real-time posture feedback into trajectory planning methods. For instance, a study proposed a trajectory planning method that considers the changing posture information of a quadruped robot during movement [2]. This method involves building kinematics equations for the mobile robot and obtaining a transformation matrix between the robot's trunk posture and foot-leg joint displacement. These are important when balancing the robot. As quadruped robot has the capability to go into the rough terrain , for this reason more and more researches are going on trajectory planning and locomotion of the robots. A dynamic kinematics model is required to administrate dynamic stability and path tracking. A research from Selçuk University showed the inverse and forward kinematics of their quadruped robot [3]. For testing and developing purposes before going straight to Hardware it is always a good practice to start with simulation for Kinematics Analysis of Robots. For Simulation purposes there are are some studies that uses Matlab,Pybullet, Python environment or ROS based Simulation Environment [4], [5]. In this study, our main goal is to derive forward kinematics and inverse kinematics and with the inverse kinematics and test the simulated result and also test it with a prototype hardware Leg.

Methodology:

1.Forward Kinematics:

The First and elementary step of doing Mathematical Calculation of a Joint enabled Robot is to make Forward Kinematics calculation. The Forward Kinematics takes the positional data in x, y, z co-ordinate and generate angles of the locomotive joints to move into that specific location of the System co-ordinate. In our Kinematics Calculation we intend to use Rotational Matrix, Translational matrix and also use Denavit-Hartenberg parameters (also called DH parameters). Using the Rotational Matrix gives an easy and also this can be used to manipulate roll, yaw and pitch of the robot's body therefore efficient conservation and manipulation techniques.



With the rotation and the translation matrices, it is possible to build a homogeneous transformation matrix along each. All individual transformation matrix involving three rotation and three translations involved within every frame of the system can be computed with the following equation,

$$M_{f-1}^{f} = M_z(\alpha)M_y(\beta)M_z(\gamma) = \begin{bmatrix} M_{11} & M_{12} & M_{13} & M_{14} \\ M_{21} & M_{22} & M_{23} & M_{24} \\ M_{31} & M_{32} & M_{33} & M_{34} \\ M_{41} & M_{42} & M_{43} & M_{44} \end{bmatrix}$$

Using The equation and D-H parameters of the Table 2.1, it is possible to create transformation matrix for each frame of the leg of the robot. Starting from the first transformation matrix, from the first joint there is a rotation of 90° along the y-axis, therefore the equation yields,

$$T_0^1 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The next translation, there is a rotation of 90° along the x-axis which translates to length La corresponding to the Coxa to Femur Joint Leg, La, in-direction to the x-axis, and a rotation of theta 1 along the z-axis, therefore the translation matrix yields,

$$T_1^2 = \begin{bmatrix} cos(\theta_1) & 0 & sin(\theta_1) & Lacos(\theta_1) \\ sin(\theta_1) & 1 & -cos(\theta_1) & Lasin(\theta_1) \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The Third transformation matrix, there is a rotation along the z axis and a translation of the leg length Lb, which is Femur to Tibia connecting Joint Leg length and the equation yields,

$$T_2^3 = \begin{bmatrix} cos(\theta_2) & -sin(\theta_2) & 0 & Lbcos(\theta_2) \\ sin(\theta_2) & cos(\theta_2) & 0 & Lbsin(\theta_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

For the Fourth transformation matrix, there is a rotation along the z axis and a translation of length Lc along the x-axis, that is Femur to End Effector Joint leg length and the equation yields,

$$T_3^4 = \begin{bmatrix} cos(\theta_3) & -sin(\theta_3) & 0 & Lccos(\theta_3) \\ sin(\theta_3) & cos(\theta_3) & 0 & Lcsin(\theta_3) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Now all these individual translational matrix can be multiplied to make a transformation matrix that defines from the leg's initial frame to the foot frame. The final matrix would obtain like equation as,

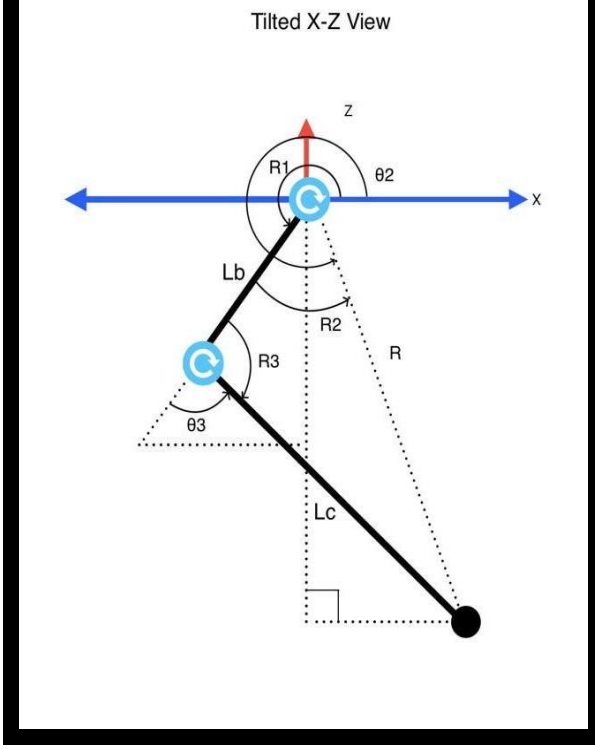
$$T_0^4 = T_0^1 T_1^2 T_2^3 T_3^4 = \begin{bmatrix} R & S \\ 0 & 1 \end{bmatrix}$$

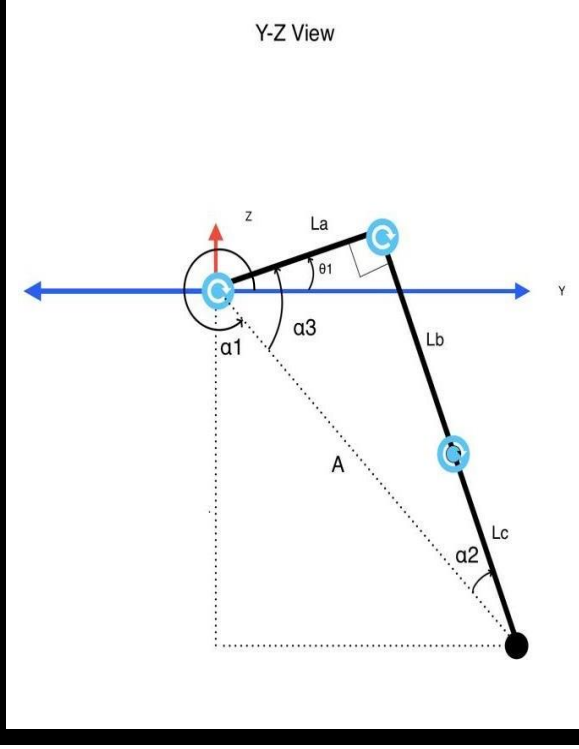
Where R is the resulting rotation matrix and S is the position matrix.

2. Inverse Kinematics Analysis: Since Forward Kinematics is a way of getting to the desired position of the co-ordinate with given angles so the backward process of getting angles from the desired co-ordinate is Inverse Kinematics. In Hardware coding it is essential to send the desired angles via as Serial Communication from Processing unit to the Micro-Processor Unit and thus the Microprocessor unit send it as PWM Signals to Motor Controller. Hence Inverse Kinematics Plays a crucial role in locomotive Robots.

Table 1: Simplified Model of Legs and Parameter for the calculation Approach

Math & Calculation: Inverse Kinematics
Known: x, y, z, rot (pitch, roll, yaw), shoulder length, wrist length Desired: $\theta_1, \theta_2, \theta_3$





In our calculation approach, we would first consider two views of one single leg from the: Simplified Model of Legs and Parameter for the calculation Approach Table. Noticeably, The Figure of the left is a tilted X-Z view, and it is tilted with respect to

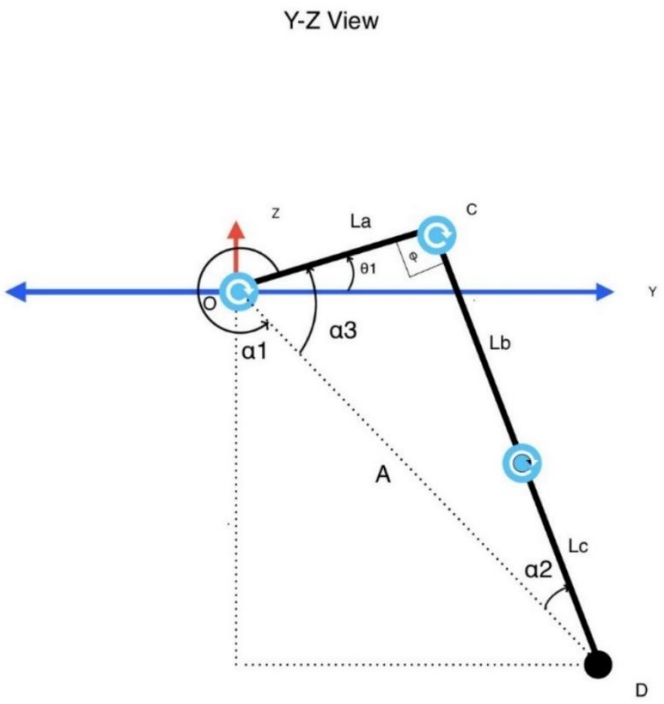


Fig. 1: Utilizing Front View of Single Robot's Leg for IK Calculation to determine Theta 1

Based on Figure 1,

Symbol	Description
La	Coxa to Femur Joint Leg length
Lb	Femur to Tibia Joint Leg length
Lc	Femur to End Effector Joint leg length

The first approach is to find Alpha 1 , therefore we connect an imaginary dotted line with respect to end-effector of the robots leg and thus creates a right angled Triangle, ΔOCD,

In, ΔOCD, $\varphi = 90°$,

A is the projection of Y-Z, so,

Therefore, $A = \sqrt{y^2 + z^2}$ ……(1)

We are considering Revolutions when measuring angles. Hence, All the rotation starts from the positive y-axis of the Cartesian Co-Ordinate System. Based on the Rotation of all the leg figures La,Lb,Lc with respect to joints , the following scenario in Fig 2 can be the case that is from 1st to 4th co-ordinate. In our Algorithm we have considered each cases along with edge cases. For each case in our Algorithm all cases have been considered and calculated equations for a_1,

$a_1 = tan^{-1}\left(\frac{y}{z}\right)$ ……(2) (Case 1 : Cartesian co-ordinate value of y and z is positive)

$a_1 = -|tan^{-1}\left(\frac{y}{z}\right)| + \pi , \pi = 180°$ ……(3) (Case 2 : Cartesian co-ordinate value of y is negative and z is positive)

$a_1 = |tan^{-1}\left(\frac{y}{z}\right)| + \pi , \pi = 180°$ ……..(4)) (Case 3: Cartesian co-ordinate value of y and z is negative)

$a_1 = -|tan^{-1}\left(\frac{y}{z}\right)| + 2\pi , \pi = 180°$…..(5) (Case 4: Cartesian co-ordinate value of y is positive and z is negative)

Case 1 and Case 3 would is applicable for Right Side Legs.Case 2 and Case 4 is for Left side legs.

For each case alpha1 is varied and therefore the theta1 value will be varied as theta 1 value caln be iterate from a_1.

In Triangle, ΔOCD,

$\alpha_2 = sin^{-1}(\frac{L_a}{A} sin\Phi)$…….(6)

$\alpha_3 = (\pi - \phi - \alpha_2) , \phi = 90°, \pi = 180°$

or, $\alpha_3 = (90 - \alpha_2)$

or, $\alpha_3 = -sin^{-1}(\frac{L_a}{\sqrt{y^2+z^2}}) + 90°$ ……(7)

$\alpha_1 = -\alpha_3 + \theta_1$, (Considering Rotation and all rotation is considered from the positive y-axis in Fig 1)

Therefore, generalized Equation for $\theta_1 = \alpha_1 + \alpha_3$, [Range 0 to 2pi],

For each iteration of cases θ_1 $value\ yields\ as\ follows$:

Case 1:

$\theta_1 = tan^{-1}\left(\frac{y}{z}\right) + (\pi - \phi - \alpha_2)$

$= tan^{-1}\left(\frac{y}{z}\right) + (90 - \alpha_2)$

$= tan^{-1}\left(\frac{y}{z}\right) + (90 - sin^{-1}(\frac{L_a}{A}\ sin\Phi\ )\ )$

$= tan^{-1}\left(\frac{y}{z}\right) + (90 - sin^{-1}(\frac{L_a}{A}\ )\ )$, [sin (90) = 1, $\Phi = 90°$]

$= tan^{-1}\left(\frac{y}{z}\right) + (90 - sin^{-1}(\frac{L_a}{A}\ )\ )$

$= \tan^{-1}\left(\frac{y}{z}\right) + 90 - sin^{-1}(\frac{L_a}{\sqrt{y^2+z^2}}\ )$ ……. (7)

Since, in our algorithm in the Range is Range 0 to 2pi and for Case 1 the
$\theta_1\ value\ would\ exceeds\ the\ range$. So the mapped equation in the algorithm yields,

$\theta_1\ =\ \tan^{-1}\left(\frac{y}{z}\right) + 90 - sin^{-1}(\frac{L_a}{\sqrt{y^2+z^2}}\ )\ -\ 2\pi$ ……(8)

For Case 4:

$\theta_1 =\ -|tan^{-1}\left(\frac{y}{z}\right)|\ +\ 2\,\pi\ + 90° - sin^{-1}(\frac{L_a}{\sqrt{y^2+z^2}}\ )\ - 2pi$ ……(9)

For Case 2:

$\theta_1\ = -|tan^{-1}\left(\frac{y}{z}\right)|\ +\ \pi\ -sin^{-1}(\frac{L_a}{\sqrt{y^2+z^2}}\ )\ +\ 90°$……(10)

For Case 3:

$\theta_1\ = |tan^{-1}\left(\frac{y}{z}\right)|\ +\ \pi\ -sin^{-1}(\frac{L_a}{\sqrt{y^2+z^2}}\ )\ +\ 90°$……(11)

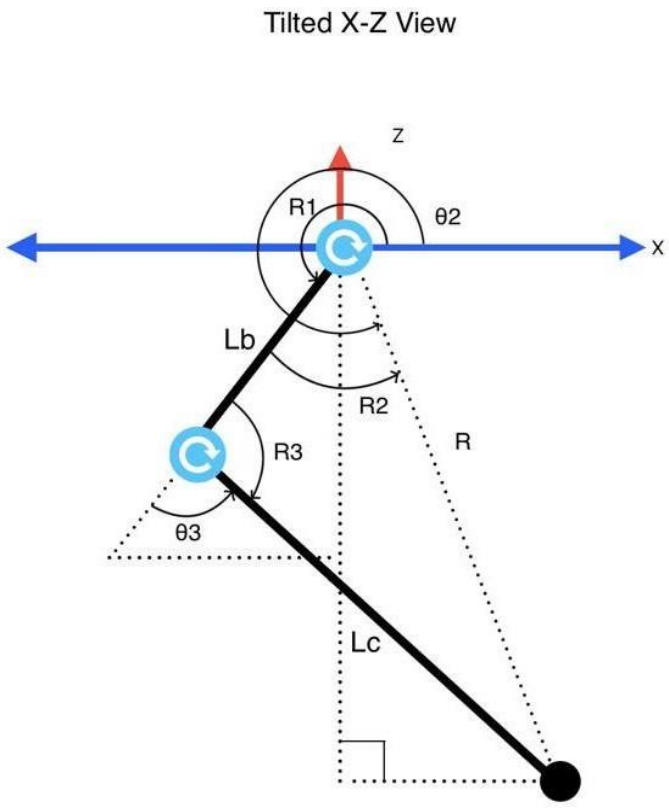


Fig. 2: Utilizing Front View of Single Robot's Leg for IK Calculation to determine Theta 2

$R = \sqrt{x^2 + z^2}$

$R_1 = tan^{-1}\left(\frac{z}{x}\right) + \pi$ (For left leg)

$R_1 = -|tan^{-1}\left(\frac{z}{x}\right)| + 2\pi$ (For right leg) …….(12)

$R_2 = cos^{-1}\left(\frac{L_b^2+R^2-L_c}{2.L_b.R}\right)$…….(13)

$R_3 = cos^{-1}\left(\frac{L_b^2+L_c^2-R^2}{2.L_b.L_c}\right)$…….(14)

$$\theta_2 = R_1 - R_2$$

Substituting and Simplifying these terms,

Substituting R1 and R2 :

$$\theta_2 = (tan^{-1}\left(\frac{z}{x}\right) + \pi) - cos^{-1}\left(\frac{L_b^2+R^2-L_c}{2.L_b.R}\right))$$

Substituting $R = \sqrt{x^2+z^2}$:

$$\theta_2 = (tan^{-1}\left(\frac{z}{x}\right) + \pi) - cos^{-1}\left(\frac{L_b^2+(\sqrt{x^2+z^2})^2-L_c}{2.L_b.R}\right))$$

Simplifying Further,

$$\theta_2 = (tan^{-1}\left(\frac{z}{x}\right) + \pi) - cos^{-1}\left(\frac{L_b^2+x^2+z^2-L_c}{2.L_b.\sqrt{x^2+z^2}}\right) \text{.....(15)}$$

$$\theta_3 = \pi - R_3$$

$$= \pi - cos^{-1}\left(\frac{L_b^2+L_c^2-R^2}{2.L_b.L_c}\right)$$

$$= \pi - cos^{-1}\left(\frac{L_b^2+L_c^2-(x^2+z^2)}{2.L_b.L_c}\right) \text{.....(16)}$$

3. Simulation and Analysis:

We have conducted our Simulation of the Robot with Full body Kinematics in Python Environment.

We have implemented our Simulation in our code to take certain value of end effectors co-ordinate (x, y, z) and correspondingly this values were sent to inverse kinematics Calculation and therefore Generating angles. In our Simulation setup we are able to simulate with all four legs and visualize those joints and generated angles also.

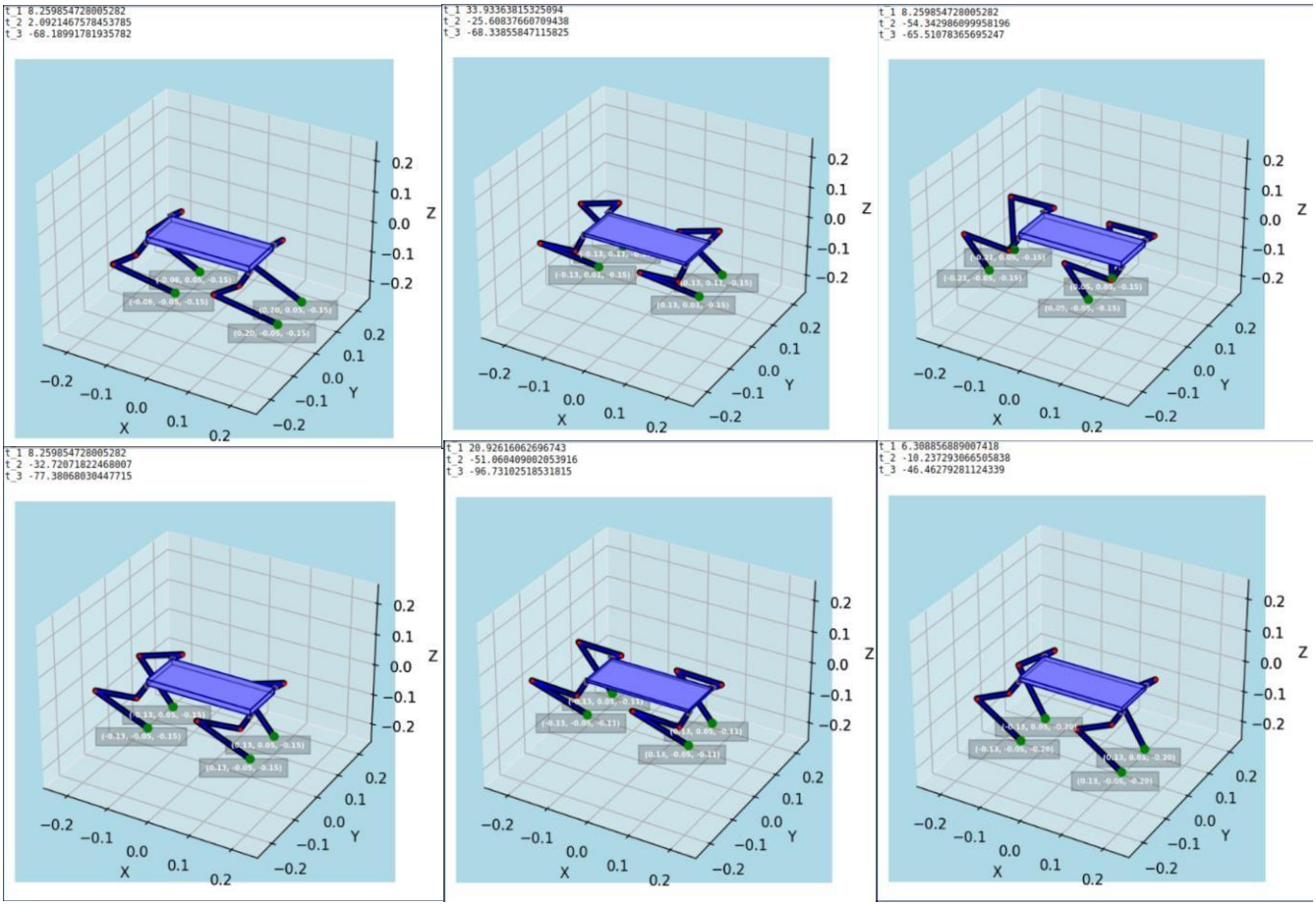


Figure : Different Pose Estimation with given co-ordinate (x, y, z) position of end effector and corresponding generated angles as t1, t2, t3 from Inverse Kinematics Calculation.

4. Control Architecture:

For Hardware testing and Control of the robot we have first experimented our leg model with a micro-controller that is connected to a computer and the communication in between is done through Serial Communication. In our Serial Communication we have also ensured Threading that way our communication protocol don't get overloaded. As per our Algorithm goes, all our calculations are done under a Class that is called the Kinematics class. Under this class there are process parameters and decision-making blocks. Robot's Parameter also plays a vital role as these values are send to Kinematiccs class for inverse kinematics calculation. Process Parameter usually takes user input for the robots leg#s end effector's position data and calculate the corresponding angles in Leg Inverse Kinematics calculator Class. Then a data Handler process all these generated angles of all four legs and send it for Data parsing and through serial communication then this datas are sent to Arduino MCU and if serial is open then to the Motor Controller unit as PWM signals for the Motors to move to the desired position.

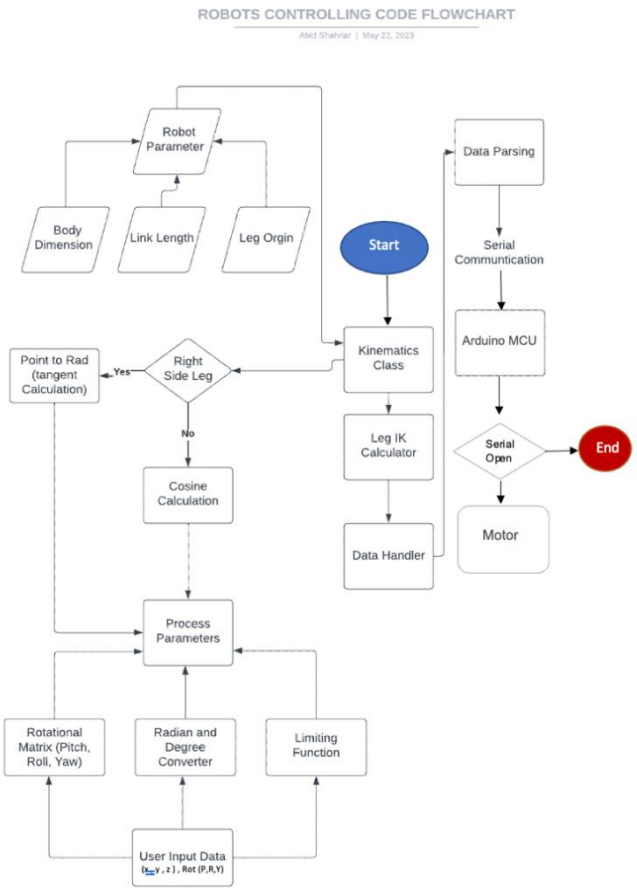


Figure: Control Architechture

Results:

After We have successfully developed our algorithm and tested it in simulation, we have interacted it with our prototype boiling leg. Through serial communication we send the angle to microcontroller and microcontroller send the signal to servo motor and we have found our prototype leg moving in accordance to our generated degree. Checking it with serial monitor we have also found the correct angle generated in our terminal resulting no error when we move the leg. In our test we have found it working as expected and therefore our prototype building leg aligns and moving with respect to the generated angles.

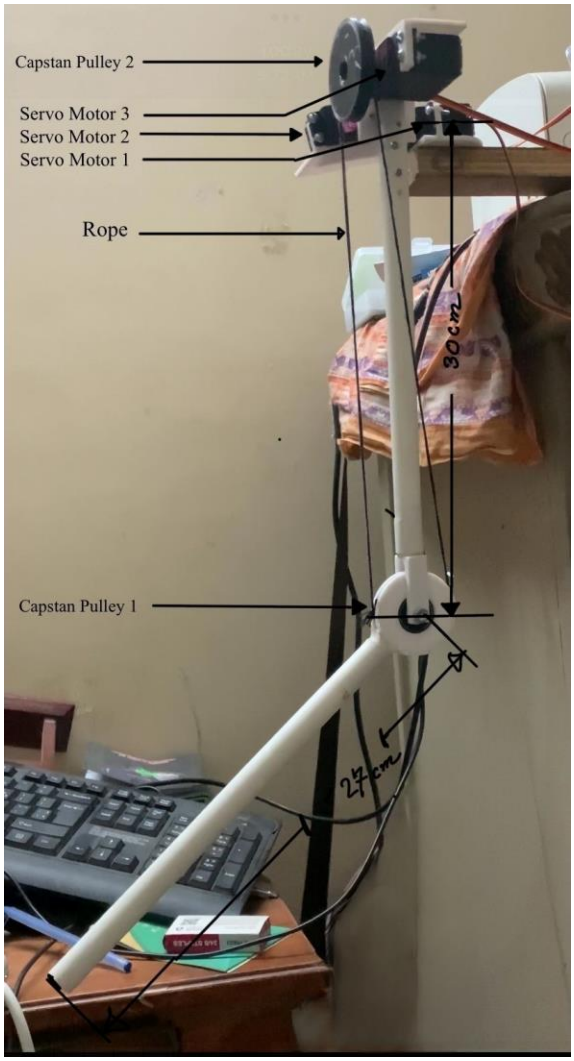


Figure: Software Interaction with Prototype Leg

Figure 6.3.2: Terminal output of generated Angles in Degrees

Conclusion: Kinematic analysis is important for developing robots, especially for bio-inspired robots . In this study, the core concentration was on a single leg of our quadruped robot and developed equations to describe its motion. In forward kinematics, we wanted to find the position and orientation of the end effector (the foot) of the leg, given the angles of the joints. To do this, we used a mathematical method called Denavit-Hartenberg parameters also using rotational matrix. We then multiplied matrices together to calculate the end effector position and orientation. After completing forward kinematics, we developed inverse kinematic equations for each joint angle of the leg. In inverse kinematics, our goal was to find the desired joint angles, given the desired end effector position and orientation using simplified trigonometric formulas also considering all different cases. We used Python and Jupyter Environment to run and simulate our inverse kinematics equation. We wrote a Python program that takes the joint angles as input and outputs the end effector position and orientation. We then used these values in the inverse kinematic equations to calculate the desired joint angles and successfully it yields correct results with mathematical calculation. To meet our goal in this paper, we derived our kinematic equation and done full body simulation of our quadruped robot, which defines the position, orientation, and movement of the robot. As we have been successful implementation in simulation In future these algorithm will help developing more advance real word quadruped robot.

Reference:

1. Rodrigo, Eng & Murillo Aranda, Rodrigo. (2022). "CONTROL OF A QUADRUPED LEG USING AN ANALYTICAL APPROACH" Thesis for obtaining the degree of Master in Optomechatronics. 10.13140/RG.2.2.29418.24006.
2. Ren, Dongyi & Cui, Yushu. (2023). Optimal Trajectory Planning Control for Quadruped Robot. Journal of Physics: Conference Series. 2587. 012067. 10.1088/1742-6596/2587/1/012067.
3. RunBin C, YangZheng C, Lin L, Jian W, Xu MH. Inverse Kinematics of a New Quadruped Robot Control Method. International Journal of Advanced Robotic Systems. 2013;10(1). doi:10.5772/55299
4. Weerakkodi, Nipun & Zhura, Iana & Babataev, Ildar & Elena, Nazarova & Fedoseev, Aleksey & Tsetserukou, Dzmitry. (2022). HyperDog: An Open-Source Quadruped Robot Platform Based on ROS2 and micro-ROS. 10.48550/arXiv.2209.09171.