ENPM662: INTRO TO ROBOT MODELLING

Term project on:

Forward and Inverse Kinematics modelling of a Quadruped robot: UMD SpotMini

Under the Guidance of:
<u>Dr. Chad Kessens</u>
University of Maryland, College Park

Submitted by:
<u>Raghav Agrawal</u>
115078055
<u>Toyas Dhake</u>
116507271

Abstract

This report is a term project for the course ENPM662: Introduction to Robot Modelling. Performed and compiled by Raghav Agarwal (115078055) Toyas Dhake (116507271) and under guidance of Prof. Chad Kessens. This report deals with the detailed analysis and modeling of UMD SpotMini based on the Boston Dynamic's SpotMini which is a quadruped robot with four legs each having three degrees of freedom and one gripper mounted on top of the chassis having five degrees of freedom.

In this report we derive the forward and inverse kinematics equations for the four legs and the gripper. The derived equations can be used to achieve any end effector position and desired orientation in space. This report further explains the gait cycle (mainly trot cycle) being used by the robot to travel from one point to the other. For walking in a straight line, the trajectory of robot's leg should be kept fixed. In order to achieve this we used the findings from our previous project of ENPM667: Control of Robotic Systems called Grey wolf optimization-based tuning of Hybrid LQR-PID controller for foot trajectory control.

Finally our aim is to perform the following tasks – the robot should be able to pick and drop object and walk in a straight path while avoiding obstacles in the path.

Table of Contents

Table of figures

Introduction

As robots are increasingly becoming an integral part of day-to-day life, there is a need for robots designed to move and act in environment designed for humans. Unlike traditional computers, robots can move around and interact with the environment. But the mobility mechanism of a robot raises problems. One of the main question that arises is what is the most efficient and reliable way for a robot to get around? For many observers, the debate comes down to an argument between wheeled robots and legged robots. The debate on robot mobility reflects a fundamental engineering challenge. Legged robots overall have some keen advantages over wheeled robots as legged robots can navigate through rough terrain and can be sent into a hazardous environment or uneven natural environments like mountains. Whereas wheeled robots lack the same ability of a legged robot to overcome different obstacles in path and requires more energy and stability when travelling through an irregular terrain. Legs can navigate through sand. They can navigate through steep hills, bombed out terrain, ledges. This ability to ascend vertical surfaces is crucial. Stepping over things, like mines or roots or battlefield debris, is also crucial. When it comes to avoiding obstacles, wheeled robots have some limitations such as if the obstacle is more than twice as high as the robot's front axle, the robot is pretty much boned. For this project I have decided to model a four legged robot similar in design to SpotMini with an arm mounted on top of the chassis. The quadruped robot has four legs with each leg having two links and three degrees of freedom and one arm having five degrees of freedom.

1.1 Motivation

Today mobile manipulation robots are a major research area in the field of robotics. These types of robots are being implemented in multitude of environments like household, health care, military operations, for study and research purposes in extreme conditions like around a nuclear disaster site. So, the mobile manipulator can be used for a large variety of purposes. A UMD SpotMini can be used to access places that are not accessible by humans or by wheeled mobile robots. Taking inspiration from biology where several mechanisms exist to perform locomotion, many legged robots are built. They are different, from their work goal to design. Bringing ideas from existing mechanisms in nature, we can develop a machine that can be used to help in search and rescue during a natural calamity.

Recent studies show that natural disasters are on the rise — dramatically so. Since 1970, according to the United Nations, the number of cataclysmic events worldwide has quadrupled. When it comes to disaster response, we need faster and safer ways to rescue people affected by the natural/man made calamities. Our UMD SpotMini is expected to successfully navigate through the debris scattered in the affected region thereby lifting heavy materials and helping people stuck under the debris.

Why use a legged robot for rescue operations?

UMD SpotMini because of its compact size can handle objects, climbs stairs, and operates in offices, homes, and outdoors. UMD SpotMini can go where larger robots cannot. Its optional arm/gripper, which attaches where a real dog's head would be, allows the legged robot to do things like open doors. Thus, the compact size of UMD SpotMini can be used to reach places where larger rescue robots are unable to access thereby increasing the speed of rescue operation.

1.2 Aim

We aim to model a robot called UMD Spot Mini, modeled after Boston dynamics' Spot Mini. Our aim is to perform forward and inverse kinematics for each of the four legs and gripper so that it can perform simple tasks like walking, picking up objects while avoiding obstacles during walking. For the robot's walking motion we decided to use trot gait as we were able to achieve maximum stability and the walking motion trajectory was optimized by Grey wolf optimization-based tuning of Hybrid LQR-PID controller for foot trajectory control.

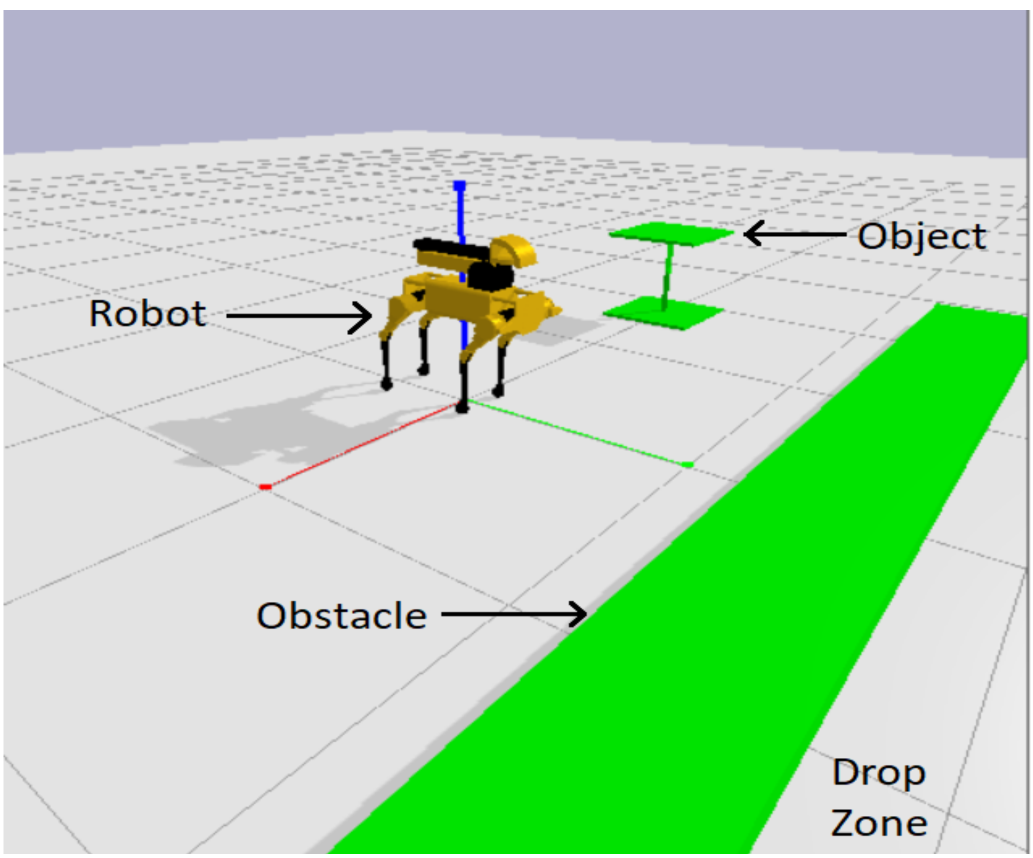


Figure 1 Simulation Environment

The figure above shows the simulation environment of the robot. There is an object dropped near the robot. The robot then picks up the object, walks towards drop zone avoiding the obstacles in path and finally drops/delivers the object in the drop zone.

Model Description

UMD SpotMini has four legs and a gripper arm attached to its chassis.

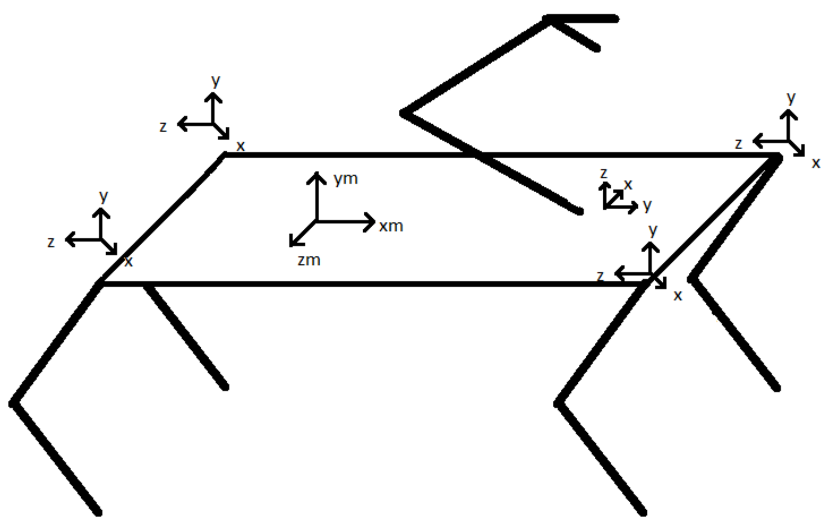


Figure 2 Structure of Robot

1. Each leg will have 2 links and 3 degrees of freedom. Exact dimension of the legs will be decided based on the CAD model. Each leg has three degrees of freedom which is the bare minimum required for the motion of UMD SpotMini in 3D space. Furthermore, three degrees of freedom are required to maintain angular span and position of stance for the robot to move effectively.

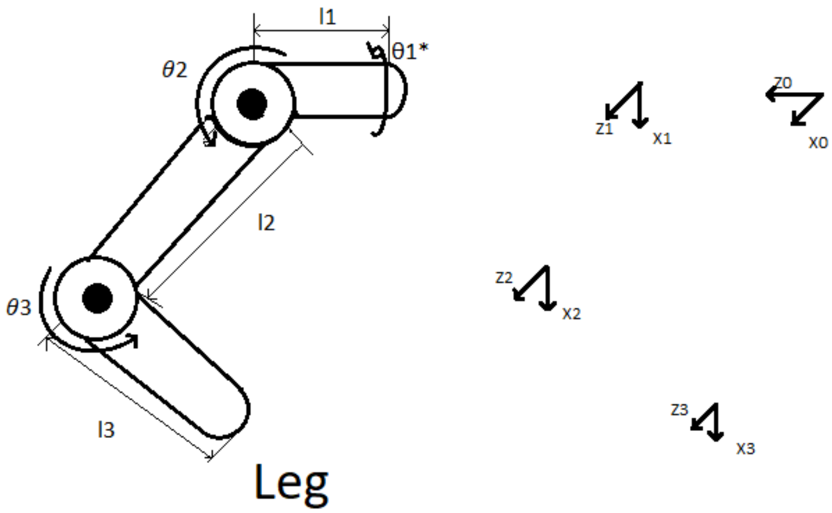


Figure 3 Structure of each leg

2. Gripper arm will have 5 degrees of freedom. Exact dimension of the arm will be decided based on the CAD model. The first degree of freedom allows the gripper to rotate about the base. The second and third degree of freedom gets the end effector to the required location. Fourth one controls end effectors orientation in conjunction to the location of the object being picked. Finally, the last degree of freedom allows the gripper to grab the object.

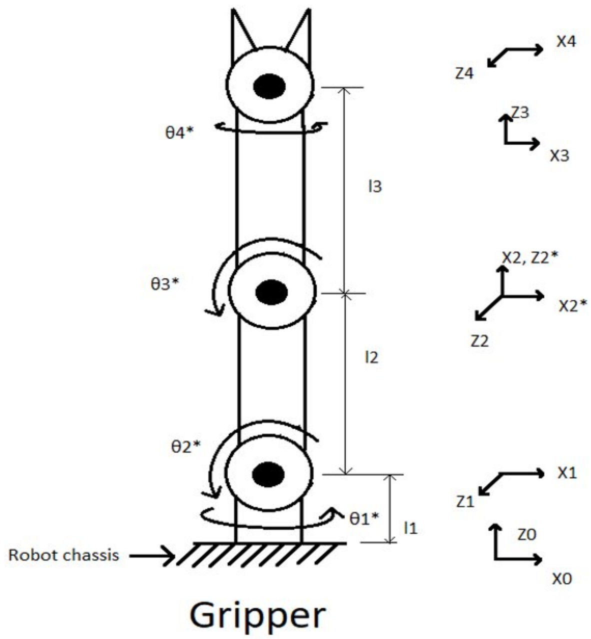


Figure 4 Structure of arm

2.1 Model Assumptions

Some assumptions are needed to simplify the simulation process. Assumption for this project are listed as follows:

1. All links are rigid, and joints are considered ideal.
2. All motors have infinite torque.
3. There is no backlash in motors.
4. There is no friction between joints.
5. There is infinite friction between the robot and ground.
6. There is infinite friction between gripper and object.
7. All the objects suspended in the environment except the robot are stationary.
8. The robot is considered to be moving in quasi-static motion.

2.2 Link Parameters and Constraints

Link lengths:

1. Width of robot: $w = 0.175m$
2. Length of robot: $l = 0.4347m$
3. Offset of hip joint: $l_1 = 0.035$
4. Length of upper leg: $l_2 = 0.25$
5. Length of lower leg: $l_3 = 0.3$
6. Offset of gripper: $x_g = 0.12$
7. Height of gripper base from robot chassis: $g_1 = 0.06$
8. Length of lower gripper arm: $g_2 = 0.35$
9. Length of upper gripper arm: $g_3 = 0.375$

Joint Constraints:

1. Hip Joint 1: -70° to 57°
2. Hip Joint 2: -17° to 230°
3. Knee Joint: -172° to 90°
4. Arm Base: -180° to 180°
5. Arm Base and Lower arm joint: 0° to 180°

6. Lower arm joint and Upper arm joint: 0° to 180°

7. Gripper base: -180° to 180°

8. Gripper: 0° to 180°

Some of the joint constraints appear to be arbitrary but they have such values due to the design of the robot. Some of robot's parts interfere with the motion of robot e.g. The upper leg is locked by an extrusion on chassis.

Scope Description and Appropriateness

The scope of the project is to build a simple analytical model of a quadruped manipulator with a gripper mounted on top of the manipulator. Based on the joint variables of the legs and gripper the forward and inverse kinematics will be derived. The equations obtained will be purely geometrical such i.e. forces and any other real-time dynamic constraints such as friction, gravity, centrifugal/centripetal forces will not be considered.

2.3 Milestones with timeline

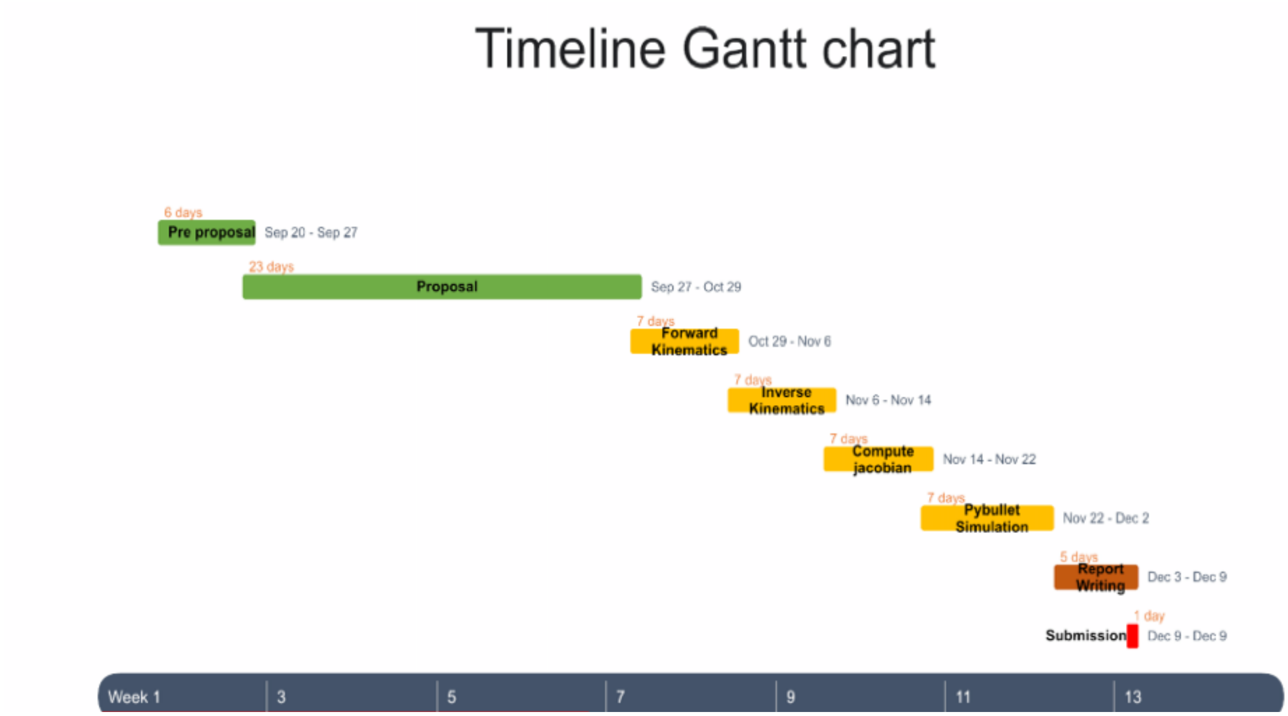


Figure 5 Gantt Chart

Task	Date of completion
Preproposal	09/27/19
Proposal	10/29/19
Forward kinematics	11/06/19
Inverse Kinematics	11/14/19
Foot trajectory mapping	11/22/19
Pybullet Simulation	12/02/19
Report Writing	12/09/19
Report Submission	12/09/19

The project was divided into different milestones as listed in the table above.
Below is the brief description of the plan of implementation:

Task 1 : Model the forward kinematics for the four legs and the gripper.

Task 2 : Model the inverse kinematics equations for the legs and the gripper.

Task 3 : Using Grey wolf optimization-based tuning of Hybrid LQR-PID controller for foot trajectory control.

Task 4 : Performing simulations to do different tasks such as picking up of an object, avoiding obstacles in path and dropping desired object at the drop zone.

Robot Modelling

The modeling of the robot starts with the computation of Forward Kinematics equation of the four legs and the gripper mounted on top of the chassis. Depending on the legs coordinates, the robot body can have different configurations. For this reason, the kinematic equation between the rotational movements (θ, φ, ψ)around the center of body's coordinate system (x_m, y_m, z_m) and the coordinate system of each endpoint of leg is investigated. Initially, to determine the position and orientation of robot's center of body in the workspace, the following transformation matrix is obtained by multiplying the rotation matrices.

3.1 Forward Kinematics

For forward kinematics we are considering the center of mass of the robot as the base frame. Hence, we must transform it from there to the base of each leg and gripper.

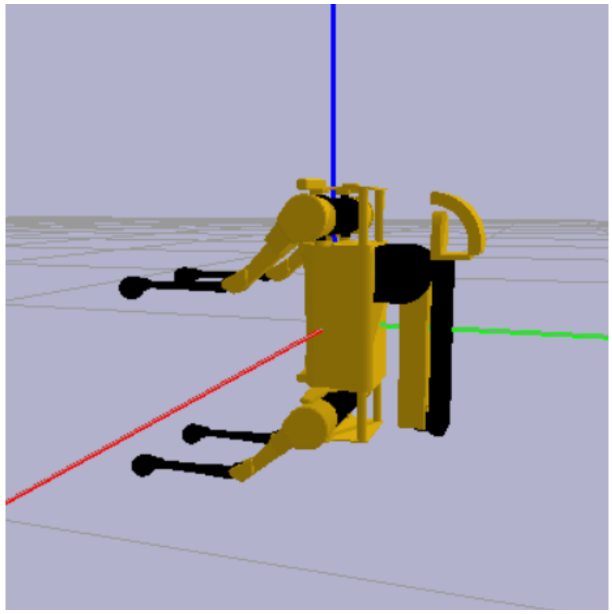


Figure 6 Initial spawn configuration of the robot

The figure shows the spawning orientation of the robot with its center of mass lying at the origin which is assumed to be the reference frame for the robot.

Rotation matrix about x-axis

$$R_x = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) & 0 \\ 0 & \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Rotation matrix about y-axis

$$R_y = \begin{pmatrix} \cos(\phi) & 0 & \sin(\phi) & 0 \\ 0 & 1 & 0 & 0 \\ \sin(\phi) & 0 & \cos(\phi) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Rotation matrix about z-axis

$$R_z = \begin{pmatrix} \cos(\psi) & -sin(\psi) & 0 & 0 \\ \sin(\psi) & \cos(\psi) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$R_{xyz} = R_x R_y R_z$$

Final transformation matrix to determine the position and orientation of the robot's center of body in the workspace

$$T_m = R_{xyz} * \begin{pmatrix} 1 & 0 & 0 & x_m \\ 0 & 1 & 0 & y_m \\ 0 & 0 & 1 & z_m \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Now using the center of body of the robot as the reference frame we can obtain the transformation matrices for all the legs and the gripper. Thus, the positions and orientations of each leg can be calculated according to the position and orientation of the robot's body.

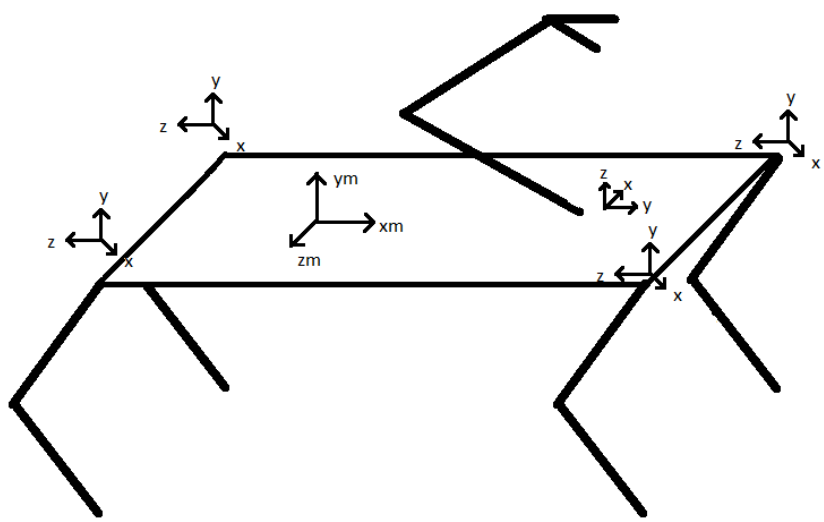


Figure 7 Base coordinate frames of the legs and the gripper

The figure above shows all the axes and orientation of all the legs and the gripper and the orientation and axis of the reference frame. Thus, the transformation matrices for the four legs and the gripper can be written as :

$$T_{right\ back} = T_m * \begin{pmatrix} \cos(\frac{\pi}{2}) & 0 & \sin(-\frac{\pi}{2}) & -\frac{L}{2} \\ 0 & 1 & 0 & 0 \\ -\sin(-\frac{\pi}{2}) & 0 & \cos(-\frac{\pi}{2}) & \frac{W}{2} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$T_{right\ front} = T_m * \begin{pmatrix} \cos(\frac{\pi}{2}) & 0 & \sin(\frac{\pi}{2}) & \frac{L}{2} \\ 0 & 1 & 0 & 0 \\ -\sin(\frac{\pi}{2}) & 0 & \cos(-\frac{\pi}{2}) & \frac{W}{2} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$T_{left\ front} = T_m * \begin{pmatrix} \cos(-\frac{\pi}{2}) & 0 & \sin(-\frac{\pi}{2}) & \frac{L}{2} \\ 0 & 1 & 0 & 0 \\ -\sin(-\frac{\pi}{2}) & 0 & \cos(-\frac{\pi}{2}) & -\frac{W}{2} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$T_{left\ back} = T_m * \begin{pmatrix} \cos(-\frac{\pi}{2}) & 0 & \sin(-\frac{\pi}{2}) & -\frac{L}{2} \\ 0 & 1 & 0 & 0 \\ -\sin(-\frac{\pi}{2}) & 0 & \cos(-\frac{\pi}{2}) & -\frac{W}{2} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$T_{gripper} = T_m * \begin{pmatrix} 1 & 0 & 0 & x_g \\ 0 & \cos(-\frac{\pi}{2}) & \sin(-\frac{\pi}{2}) & 0 \\ 0 & \sin(-\frac{\pi}{2}) & \cos(-\frac{\pi}{2}) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Where T_m is the transformation matrix to determine the position and orientation of the robot's center of body in the workspace.

3.2 Frame Assignment

Right handed coordinates were used for the axis assignment of the frames of the robot. And frames were assigned in such way that they satisfy the following DH parameter conditions:

1. Axis x_n intersects z_{n-1} axis.

2. Axis x_n is perpendicular to z_{n-1} axis. Following figure x and y shows the frame assignment used to create the legs and the gripper for UMD SpotMini.

3.3 DH table for leg

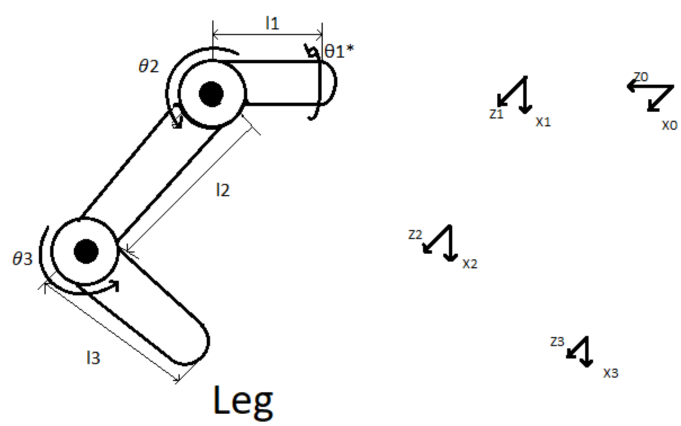


Figure 8 Co-ordinate frames for each joint (leg)

FRAMES	θ	d	a	α
0-1	θ_1^*-90^0	l_1	0	-90^0
1-2	θ_2^*	0	l_2	0
2-3	θ_3^*	0	l_3	0

3.4 Transformation Matrices for leg

Using standard form of DH matrix and DH table calculated above, transformation matrices for all frames are calculated as shown below:

$$A_1 = \begin{bmatrix} \sin(\theta_1) & 0 & \cos(\theta_1) & 0 \\ -\cos(\theta_1) & 0 & sin(\theta_1) & 0 \\ 0 & -1 & 0 & l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_2 = \begin{bmatrix} \cos(\theta_2) & -\sin(\theta_2) & 0 & l_2\cos(\theta_2) \\ sin(\theta_2) & \cos(\theta_2) & 0 & l_2\sin(\theta_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_3 = \begin{bmatrix} \cos(\theta_3) & -\sin(\theta_3) & 0 & l_3\cos(\theta_3) \\ sin(\theta_3) & \cos(\theta_3) & 0 & l_3\sin(\theta_3) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

T_3^0

$$= \begin{bmatrix} \cos(\theta_2+\theta_3)\,sin\theta_1 & -\sin(\theta_2+\theta_3)\,sin\theta_1 & cos\theta_1 & sin\theta_1[l_3\cos(\theta_2+\theta_3)+l_2cos\theta_2] \\ -\cos(\theta_2+\theta_3)\,cos\theta_1 & \sin(\theta_2+\theta_3)\,cos\theta_1 & sin\theta_1 & -cos\theta_1[l_3\cos(\theta_2+\theta_3)+l_2cos\theta_2] \\ -\sin(\theta_2+\theta_3) & -\cos(\theta_2+\theta_3) & 0 & l_1-l_3\sin(\theta_2+\theta_3)-l_2sin\theta_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

All the legs have same dimension and are identical. Hence we can use the computed forward kinematics to obtain the coordinates of the leg in the end effector frame. The coordinates are obtained by the following transformation:

Right back- $T_{right\ back} * T_3^0$

Right front- $T_{right\ front} * T_3^0$

Left back- $T_{left\ back} * T_3^0$

Left front- $T_{left\ front} * T_3^0$

3.5 DH Table for gripper

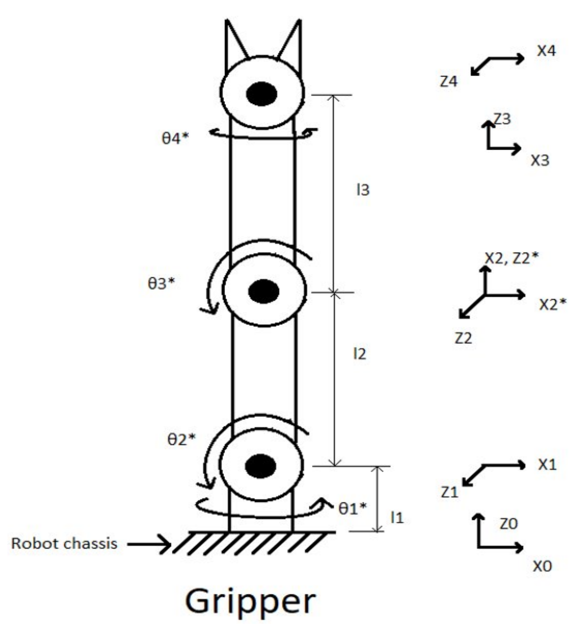


Figure 9 Co-ordinate frames for each joint (gripper arm)

Frames	θ	d	a	α
0-1	θ_1^*	l_1	0	90^0
1-2	$\theta_2^*+90^0$	0	l_2	0
2-2*	$\theta_3^*-90^0$	0	0	-90^0
2*-3	0	l_3	0	0
3-4	θ_4^*	0	0	90^0

3.6 Transformation Matrices for gripper

Using standard form of DH matrix and DH table calculated above, transformation matrices for all frames are calculated as shown below:

$$A_1 = \begin{bmatrix} \cos(\theta_1) & 0 & \sin(\theta_1) & 0 \\ sin(\theta_1) & 0 & -cos(\theta_1) & 0 \\ 0 & 1 & 0 & l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_2 = \begin{bmatrix} -\sin(\theta_2) & -\cos(\theta_2) & 0 & -l_2\sin(\theta_2) \\ cos(\theta_2) & -\sin(\theta_2) & 0 & l_2\cos(\theta_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_3 = \begin{bmatrix} \sin(\theta_3) & 0 & \cos(\theta_3) & l_3\cos(\theta_3) \\ -cos(\theta_3) & 0 & \sin(\theta_3) & l_3\sin(\theta_3) \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_4 = \begin{bmatrix} \cos(\theta_4) & 0 & \sin(\theta_4) & 0 \\ sin(\theta_4) & 0 & -\cos(\theta_4) & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_4^0 = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix}$$

m_{11}	$\sin\theta_1 sin\theta_4 + cos\theta_4 cos\theta_1 \cos(\theta_2 + \theta_3)$
m_{12}	$-\sin(\theta_2 + \theta_3)\, cos\theta_1$
m_{13}	$cos\theta_4 sin\theta_1 + sin\theta_4 cos\theta_1 \cos(\theta_2 + \theta_3)$
m_{14}	$-cos\theta_1[(\mathrm{l}_3)\sin(\theta_2 + \theta_3) + l_2 sin\theta_2]$
m_{21}	$cos\theta_1 sin\theta_4 + cos\theta_4 sin\theta_1 \cos(\theta_2 + \theta_3)$
m_{22}	$-\sin(\theta_2 + \theta_3)\, sin\theta_1$
m_{23}	$sin\theta_4 sin\theta_1 \cos(\theta_2 + \theta_3) - cos\theta_1 cos\theta_4$
m_{24}	$-sin\theta_1[(l_3)\sin(\theta_2 + \theta_3) + l_2 sin\theta_2]$
m_{31}	$\sin(\theta_2 + \theta_3)\, cos\theta_4$
m_{32}	$\cos(\theta_2 + \theta_3)$
m_{33}	$\sin(\theta_2 + \theta_3)\, sin\theta_4$
m_{34}	$l_1 + (l_3)\cos(\theta_2 + \theta_3) + l_2 cos\theta_2$
m_{41}	0
m_{42}	0
m_{43}	0
m_{44}	1

Similarly, here again we can use the computed forward kinematics to obtain the coordinates of the arm in the end effector frame.

3.7 Inverse Kinematics

We need to find inverse kinematics equations for both, legs and the gripper. After obtaining the transformation matrices and forward kinematics matrices required for the inverse kinematic solution of the Quadruped Robot, the inverse kinematic analysis is performed using analytical methods. Equations expressing the angular positions of the joints of the legs and the gripper are obtained.

Geometrical approach for computing inverse kinematics equations for leg

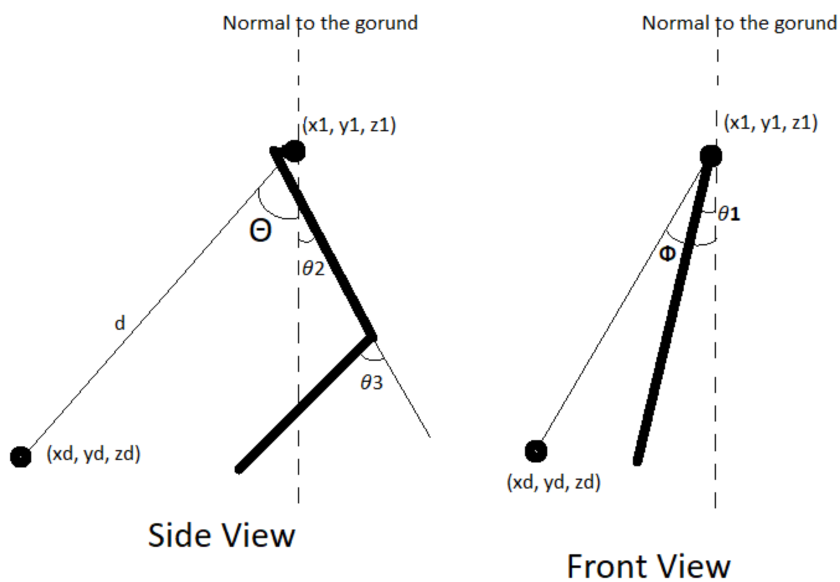


Figure 10 Inverse kinematics(leg)

The figure above shows side view and front view of the leg. In the figure above parameter d is the absolute distance between base of the leg and desired end effector coordinate. Thus d can be computed as:

$$d = \sqrt{(x_d - x_1)^2 + (y_d - y_1)^2 + (z_d - z_1)^2}$$

Φ is the angle between leg base and desired end effector point with normal to the ground in Y-Z plane. Hence, we need to introduce an angle θ_1 which is the angle between the normal and shifted leg orientation.

$$\theta_1 = atan2\left(\frac{z_d - z_1}{y_d - y_1}\right)$$

θ_2 is angle between leg base and desired point with normal to the ground from which the inner angle of triangle made by upper leg and lower leg is subtracted.

$$\theta_2 = atan2\left(\frac{x_d - x_1}{y_d - y_1}\right) - acos\left(\frac{d^2 + l_2^2 - l_3^2}{2dl_2}\right)$$

θ_3 can be easily calculated by cosine rule of triangle as we have all sides of triangle. Since θ_3 is an external angle hence we need to subtract calculated angle from 180°.

$$\theta_3 = 180 - acos\left(\frac{l_2^2 + l_3^2 - d^2}{2l_2 l_3}\right)$$

These same equations will apply for all the legs as they have same dimension and orientation.

Geometrical approach for computing inverse kinematics equations for gripper arm

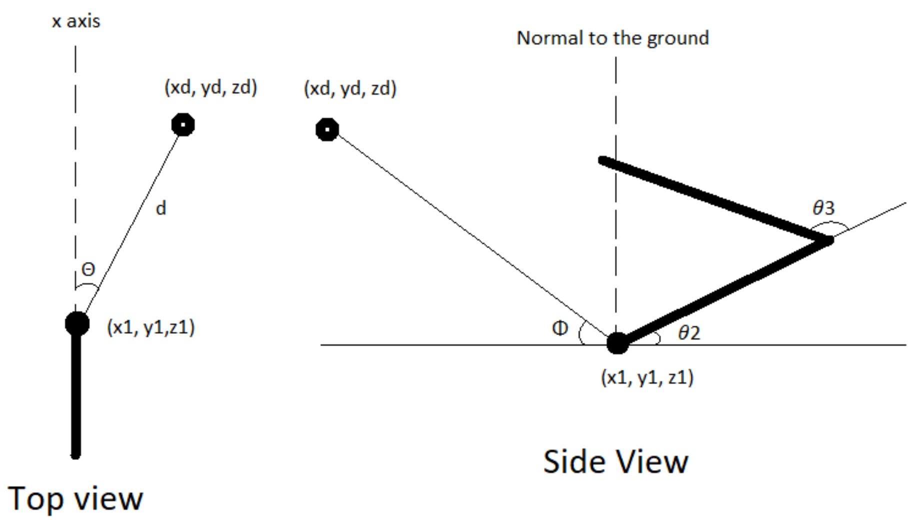


Figure 11 Inverse kinematics(gripper arm)

The figure above shows side view and front view of the leg. In the figure above parameter d is the absolute distance between base of the leg and desired end effector coordinate. Thus d can be computed as:

$$d = \sqrt{(x_d - x_1)^2 + (y_d - y_1)^2 + (z_d - z_1)^2}$$

Θ is the angle m between base of gripper arm to the desired point with X axis in X-Z plane. Hence, we can directly equate θ_1 with that angle.

$$\theta_1 = atan2\left(\frac{z_d - z_1}{x_d - x_1}\right)$$

Φ is the angle between gripper arm base to desired point with parallel to ground. Hence, to compute θ_2 we need to subtract Φ and inner angle of triangle made by upper and lower arm from 180°

$$\theta_2 = 180 - [atan2\left(\frac{y_d - y_1}{x_d - x_1}\right) + acos\left(\frac{d^2 + l_2^2 - l_3^2}{2dl_2}\right)]$$

θ_3 can be easily calculated by cosine rule of triangle as we have all sides of triangle.

$$\theta_3 = acos\left(\frac{l_2^2 + l_3^2 - d^2}{2l_2 l_3}\right)$$

θ_4 depends on the orientation of the object being picked and it has no rotational constraints and $\theta_5$controls the gripping action. It varies between 0^0 and 180^0.

3.8 Validation process

1. **Validation of inverse kinematics for the four legs**

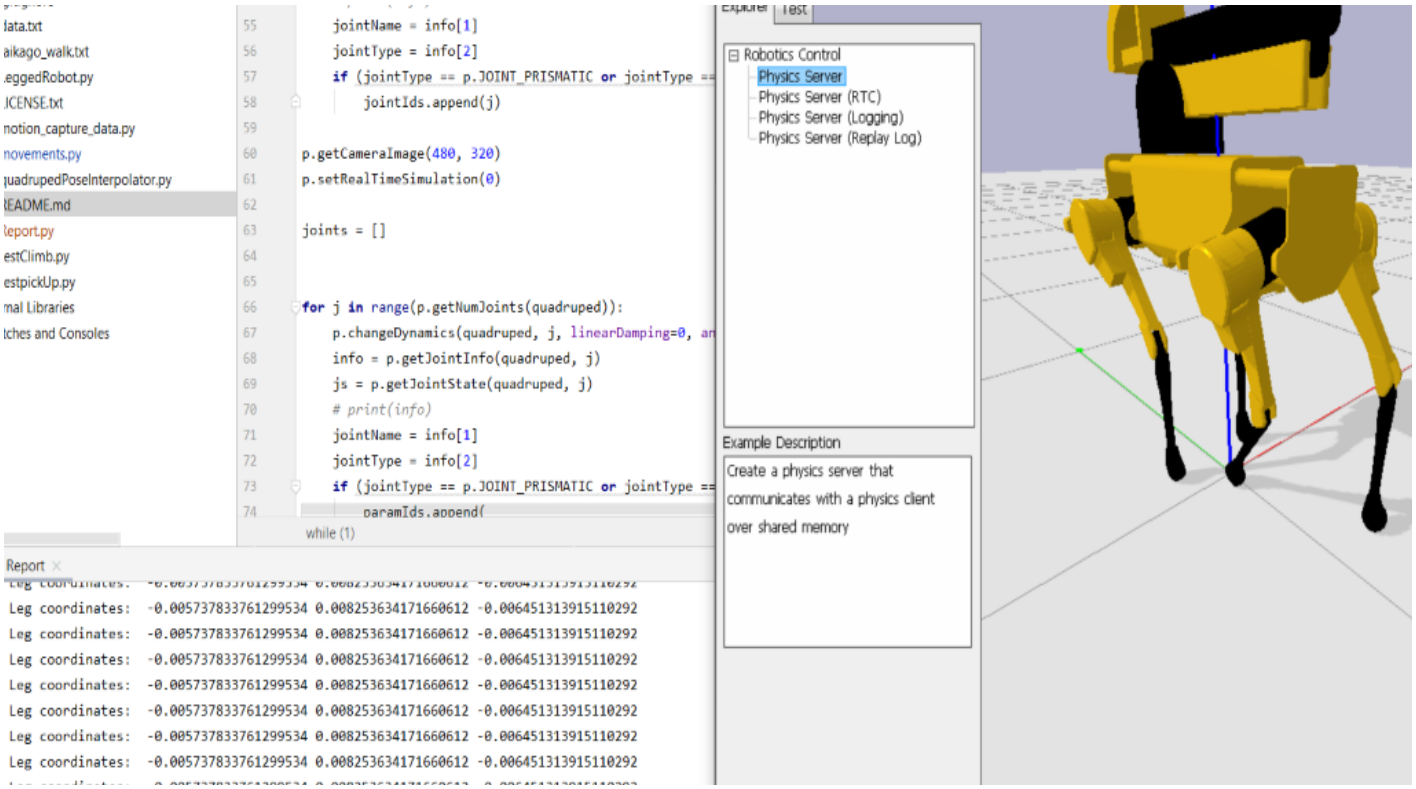
If the robot is spawned at the origin, the center of mass lies at (0,0.5,0) and the coordinates of the rear left leg is (-0.2,0.5,-0.08). Now we will compute θ_1, θ_2 and θ_3 for the leg when it is positioned at the desired end effector coordinates i.e. (0,0,0).

$$\theta_1 = atan2\left(\frac{0 - (-0.08)}{0 - 0.5}\right) = 9.902^0$$

$$d = \sqrt{(0 - (-0.2))^2 + (0 - 0.5)^2 + ((0 - (-0.08)^2} = 0.544$$

$$\theta_2 = atan2\left(\frac{0 - (-0.2)}{0 - 0.5}\right) - acos\left(\frac{0.544^2 + 0.25^2 - 0.3^2}{2 * 0.544 * 0.25}\right) = -31.086^0$$

$$\theta_3 = 180 - acos\left(\frac{0.25^2 + 0.3^2 - 0.544^2}{2 * 0.25 * 0.3}\right) = 17.013^0$$



From the figure above it can be seen that the coordinates for the leg calculated in the simulation lies close to (0,0,0) , thereby validating our results.

2. Validation of inverse kinematics for gripper

Similarly here again we will pick arbitrary end effector coordinates for the gripper arm. Let the arbitrary point be (0.5,0.5,0.5) and based on this the base coordinates of the gripper will be at $x + x_g$ i.e. (0.12, 0.5, 0). We will know compute θ_1, θ_2 and θ_3 for this position.

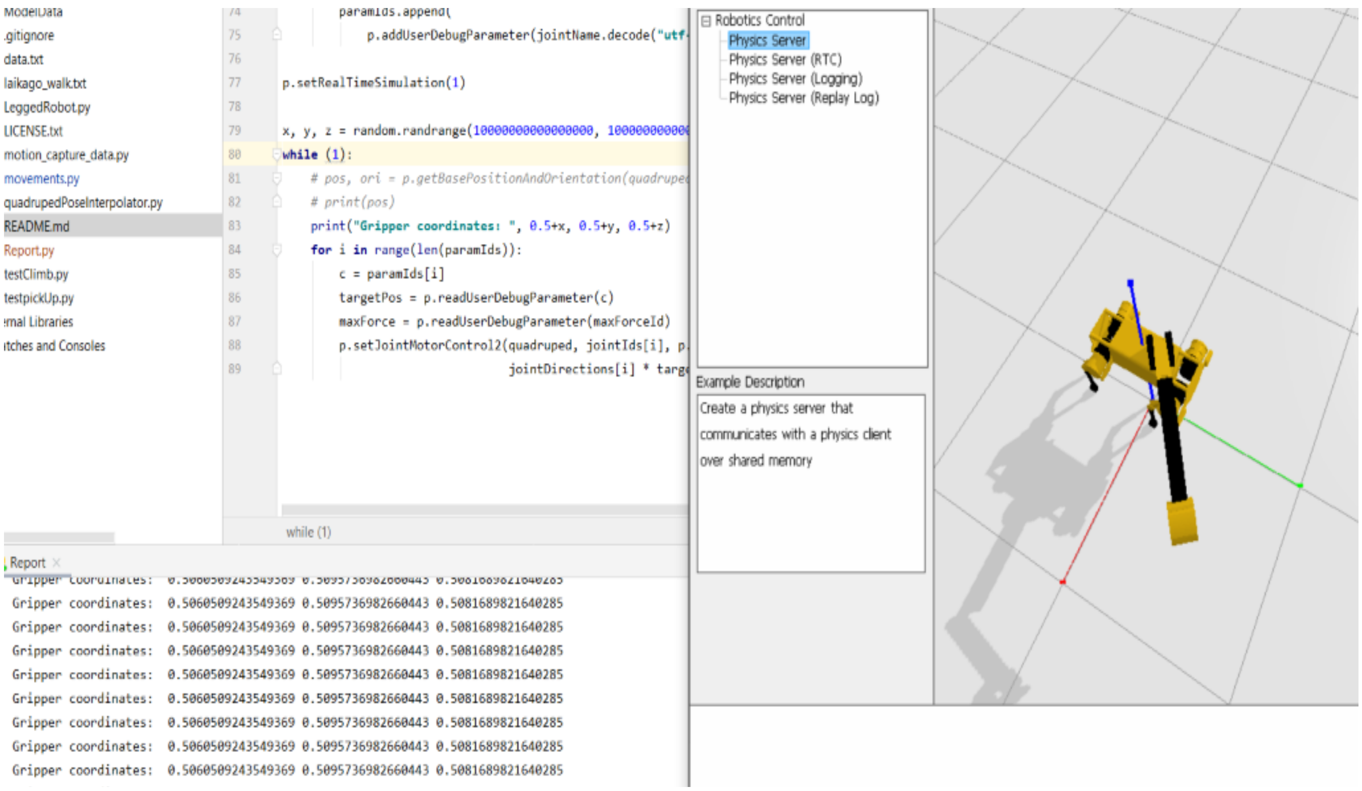
$$d = \sqrt{(0.5-0.12)^2 + (0.5-0.5)^2 + (0.5-0)^2} = 0.628$$

$$\theta_1 = atan2\left(\frac{0.5-0}{0.5-0.12}\right) = 52.765^0$$

$$\theta_2 = 180 - [atan2\left(\frac{0.5 - 0.5}{0.5 - 0.12}\right) + acos\left(\frac{0.628^2 + 0.35^2 - 0.375^2}{2 * 0.35 * 0.628}\right)]$$

$$\theta_2 = 148.861^0$$

$$\theta_3 = 180 \; - acos\left(\frac{0.35^2 + 0.375^2 - 0.628^2}{2 * 0.35 * 0.375}\right) = 120^0$$



From the figure above it can be seen that the coordinates for the gripper calculated in the simulation lies close to (0.5,0.5,0.5) , thereby validating our results.

3.9 Walking Gait Cycles

Walking gait cycles can be categorized in following types:

3. Walk – This is the slowest gait cycle. In walking gait one of the legs is always off the ground while the remaining three legs maintain contact with the surface. In walking gait the support polygon formed is a triangle. This results in maximum stability as it is easier to keep the center of mass of the body inside this triangular support polygon.
4. Trot – In this gait at any instant two diagonally opposite legs are off the ground while under two legs maintain contact with the surface. The support polygon for a trot gait is line and is smaller as compared to the walk gait. This results in decreased stability of the body.
5. Gallop – This is the fastest gait cycle that can be implemented in our robot. In this gait cycle the legs use the ground to launch itself further in air and the legs loose contact from ground. This results in fastest possible motion. In order to achieve this gait we need to implement a controller in robot to stabilize the robot. As controls is not in the scope of this project, we will not investigate this gait cycle.

Walk Vs Trot

As mentioned above walk gait is more stable when compared with trot. But walk gait cycle is slower as compared to trot gait. Initially when we implemented walk gait for robot, our robot couldn't operate in one straight line. One of the main reasons for this behavior was due to increased weight added by the gripper mounted on top of the chassis. This increase in weight resulted in shifting of the center of mass of the robot. The center of mass was found to be lying outside the support polygon thereby rendering the system unstable and preventing the robot to travel in a straight path.

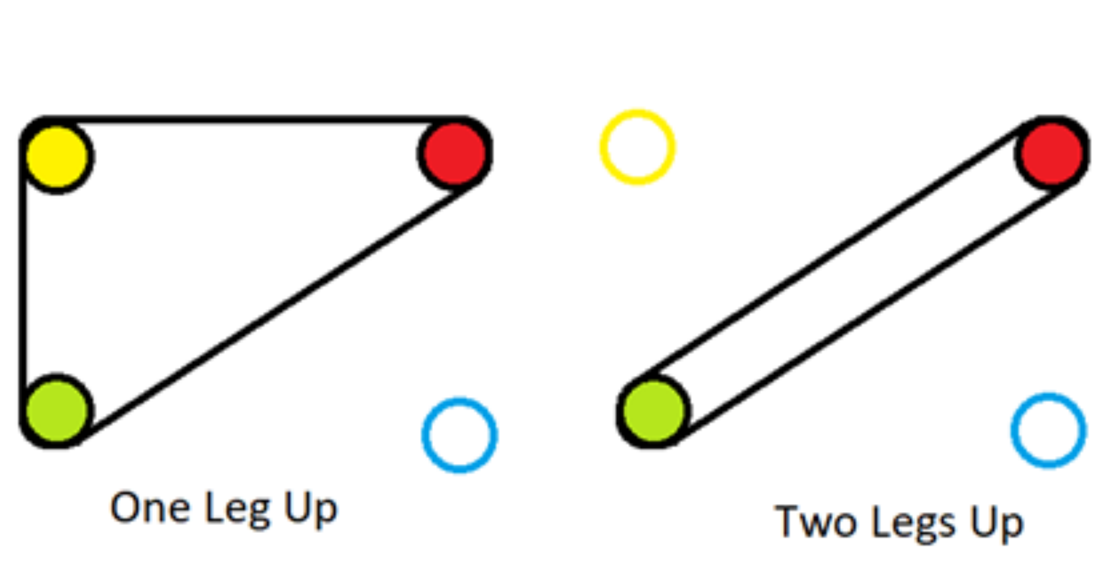


Figure 12 Support polygon for walk vs trot gait

In trot gait the two diagonally opposite legs are in air at one instant of time, this results in instability. Before running our simulation we assumed that if the motion was too slow this will result in center of mass lying outside the support polygon. In order to combat this problem we increased the speed between two successive trot cycles. Along with this we implemented our findings from our other project on Grey Wolf Optimizer Based Tuning of a Hybrid LQR-PID Controller for Foot Trajectory Control of a Quadruped Robot for controls class to control the trajectory of the robot's leg, so that it follow a semicircle trajectory.

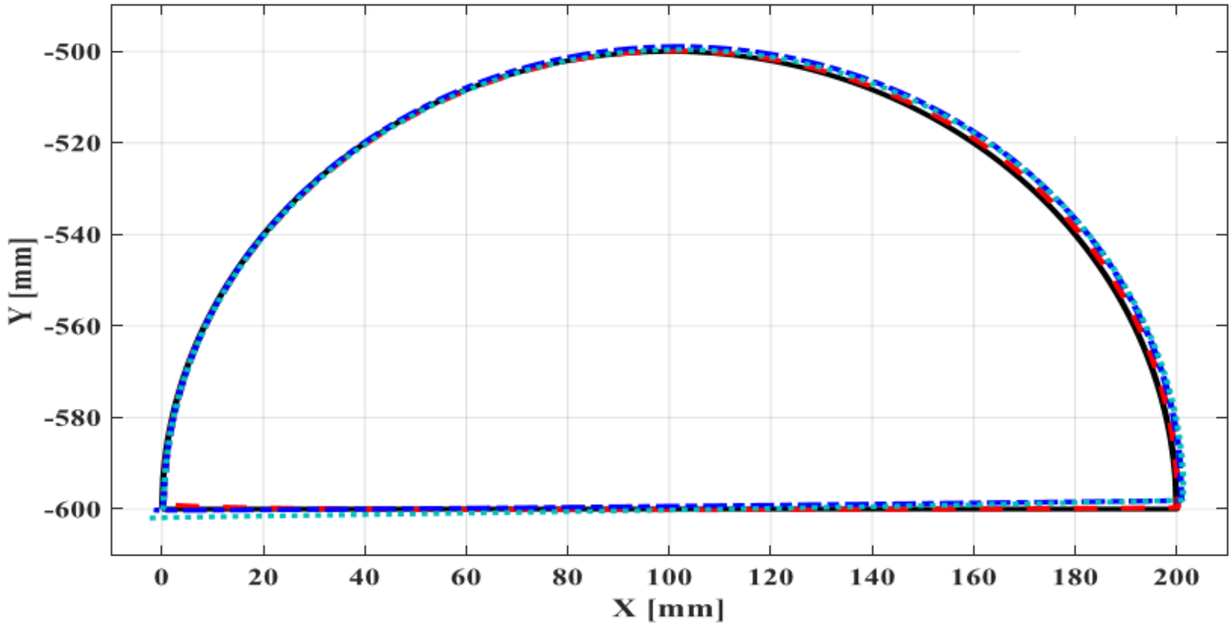


Figure 13 Foot Trajectory

The scale of the robot leg used in the simulation was different hence values on axis are different but the results were scaled accordingly. Thus, we were able to optimize the trajectory of the foot to move in a semi-circle fashion. Adjacent legs are always at opposite point on the curve. So, when one leg is in contact with the ground, the other one is in air moving forward. This cycle repeated every 500 milliseconds which means robot takes 2 steps per second. This cycle time resulted in increased stability during motion of the robot and changes in speed would result in instability.

3.10 Picking and Climbing

To implement picking up of an object we decided to use a dumbbell shaped object which was easy to grab as the object won't fall once picked up by the gripper. This particular shape of the object was implemented in the simulation as contact modeling is not in the scope of this project. Moreover, to get on top of the platform in the path, trot gait with increased speed was implemented. This resulted in faster motion of the front legs and the rear legs to assist the robot to get on top of the platform.

Simulations and Results

Based on the equations computed above, the robot is capable of performing all the desired tasks. Following are the screenshots of the simulation showing the different tasks performed by the robot.

Task 1: Use inverse kinematics to get to the location of the object and grip it.

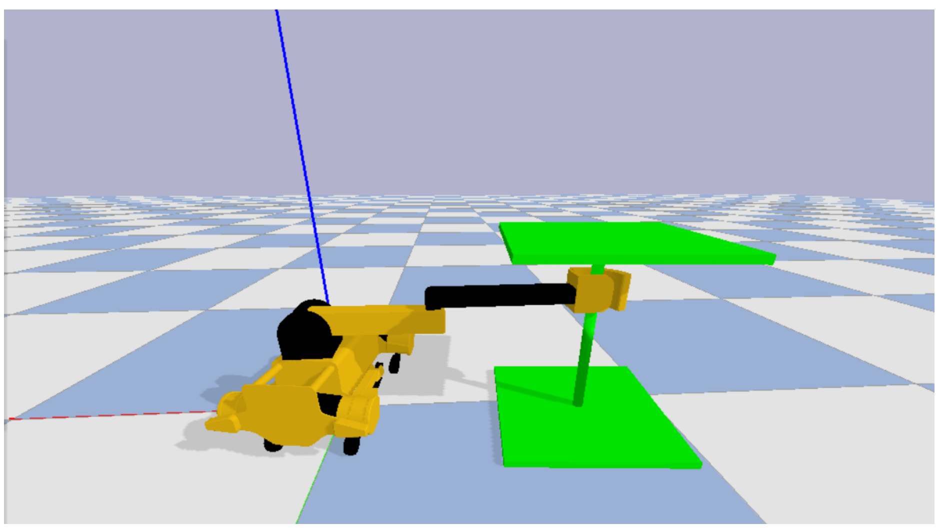


Figure 14 Gripping up of object

Task 2: Reset to original position.

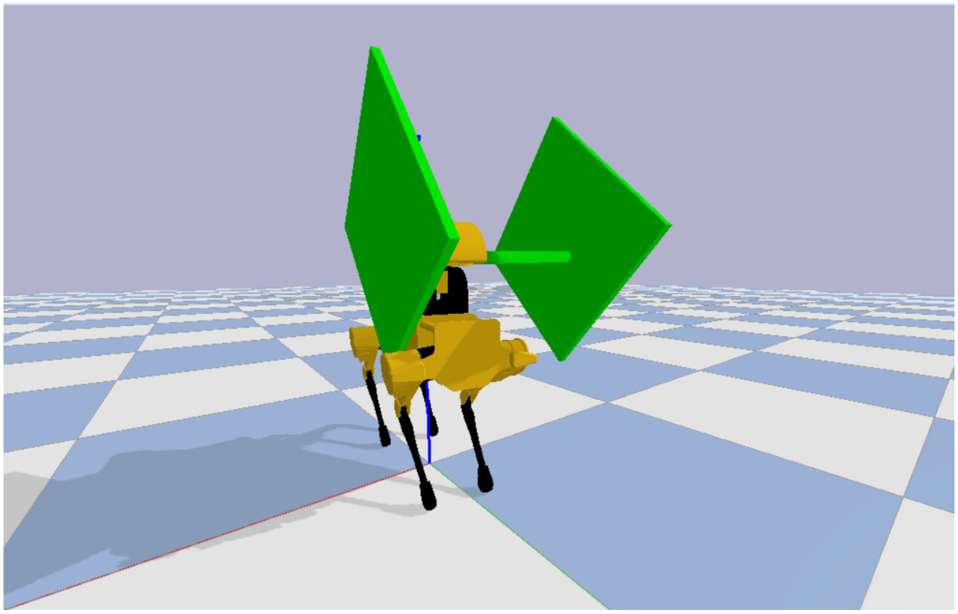


Figure 15 Moving to drop zone

Task 3: Start walking till the obstacle is encountered in the path.

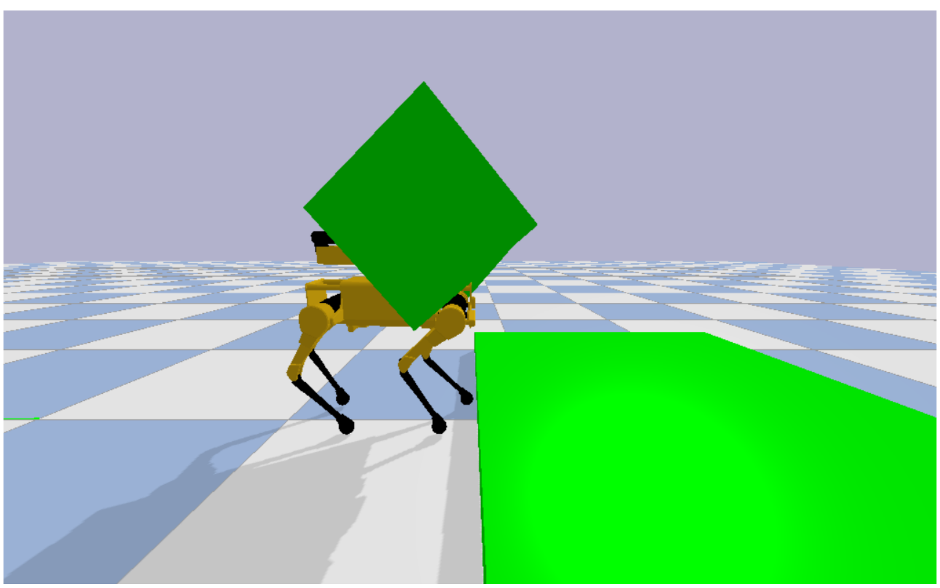


Figure 16 Obstacle detected

Task 4: Avoid the obstacle by climbing and walking over the platform.

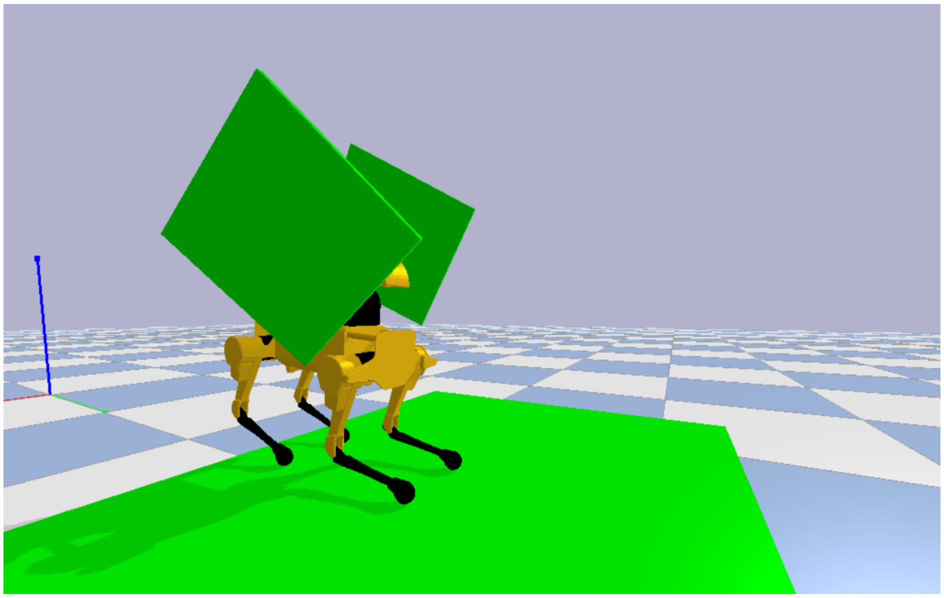


Figure 17 Climb obstacle

Task 5: Walk until the edge of the platform is reached.

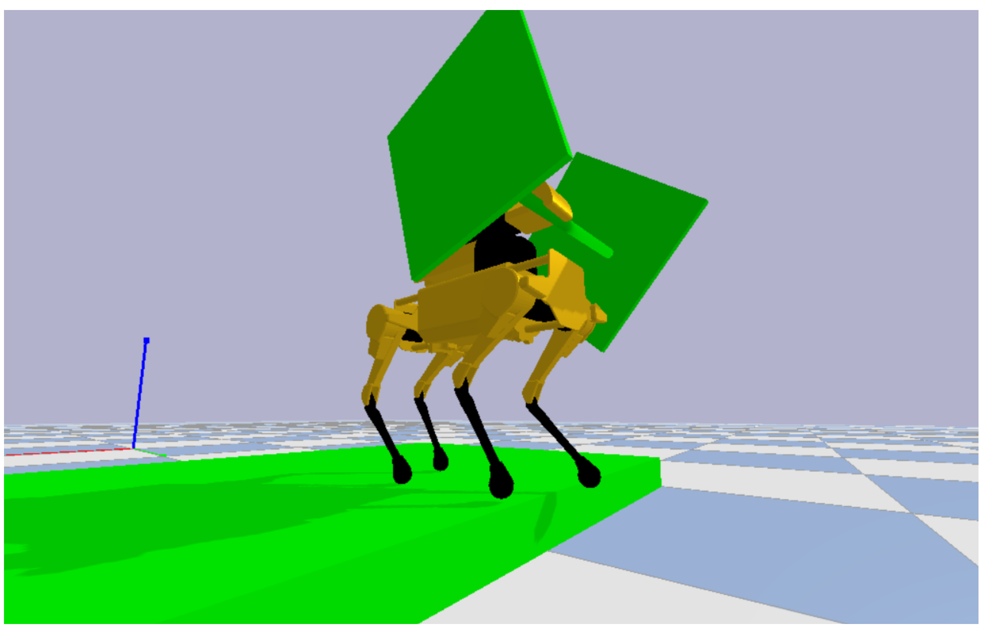


Figure 18 Walking until edge is reached

Task 6: Getting down from the platform maintaining its stability.

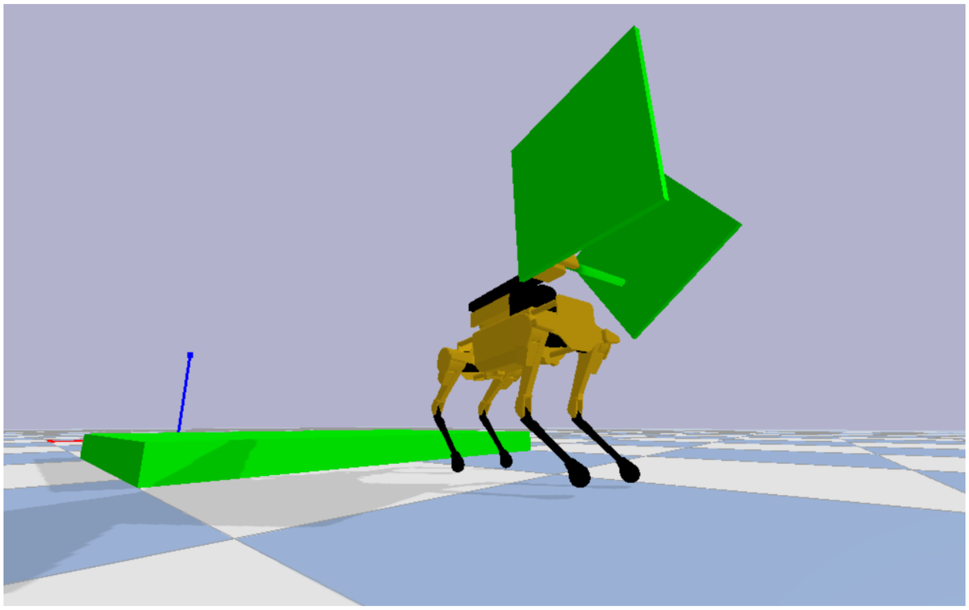


Figure 19 Getting down from platform

***Task** 7: Drop the object in the drop zone.*

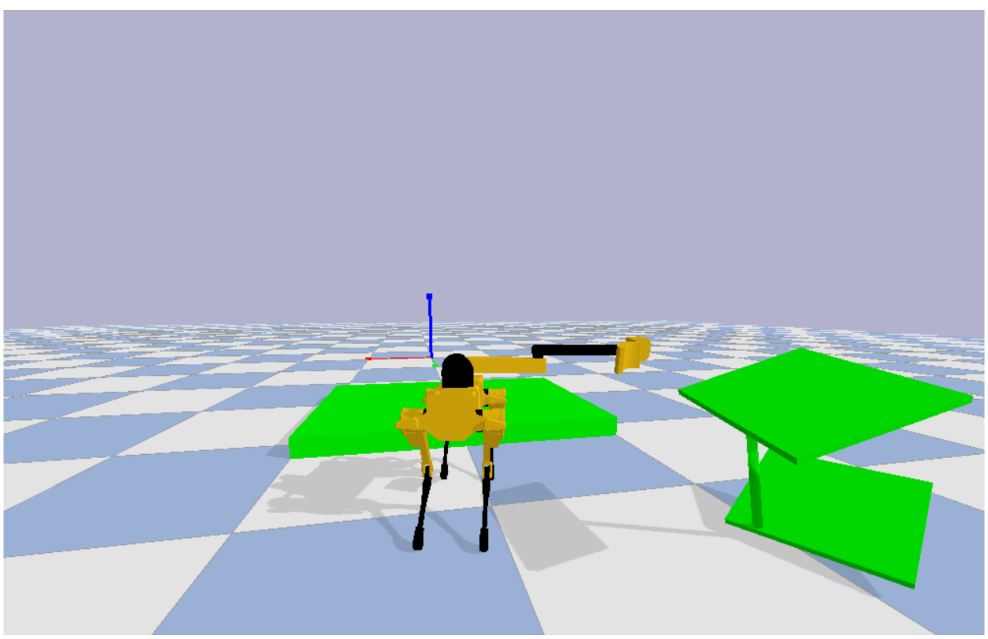


Figure 20 Object dropped in drop zone

The video of complete simulation showing all the tasks performed is submitted along with the report, also the link to GitHub repository which has files of simulation implementation, is mentioned below in Appendix.

Conclusion and Future scope

In this report we investigated the forward and inverse kinematics of a quadruped robot , UMD SpotMini modelled after Boston Dynamics' quadruped robot SpotMini. The aim of this project was to model a quadruped robot which can be used to pick and transport objects from point A to point B avoiding obstacles in path and operate in conditions where human cannot go like biological and nuclear disaster sites. The simulation was performed in Pybullet using Unitree's Laikago as base model, the model was modified in SoildWorks to add a gripper. After implementing our forward and inverse kinematics equations in our simulation we encountered some challenges. In order to stabilize the robot, the speed of the robot was locked in our simulation. Increasing or decreasing the speed results in unstable walking gait cycle. The other problem encountered was with Pybullet's collision detection mechanism, it should be avoided from being used if the sole purpose of project is contact modeling and grapping until more functions are add to it in future updates.

The UMD SpotMini can be implemented in numerous ways. We can add more gripper to the chassis in order to pick multiple objects at once. Different sensors can be mounted on the robot to make it more interactive and human friendly. We can also develop mechanisms to achieve different walking gait cycles like running, galloping. Lastly we can implement jumping mechanism for our robot.

Finally, I would like to thank Prof. Chad Kessens for allowing me to work on this project and giving me insights to complete the project and other deliverables on time, TAs for helping and supporting me throughout the course. Lastly, I would like to thank my project partner Toyas Dhake for always being supportive and helpful.

Bibliography

[1] Pubullet: https://pypi.org/project/pybullet/

[2] Laikago: https://robots.ieee.org/robots/laikago/

[3] Grey Wolf Optimizer Based Tuning of a Hybrid LQR-PID Controller for Foot Trajectory Control of a Quadruped Robot: https://www.researchgate.net/publication/335015963_Grey_Wolf_Optimizer_Based_Tuning_of_a_Hybrid_LQR-PID_Controller_for_Foot_Trajectory_Control_of_a_Quadruped_Robot

[4] Controls Project on Grey Wolf Optimizer Based Tuning of a Hybrid LQR-PID Controller for Foot Trajectory Control: https://github.com/ToyasDhake/controls_project

[5] Inverse Kinematic Analysis Of A Quadruped Robot: https://www.researchgate.net/publication/320307716_Inverse_Kinematic_Analysis_Of_A_Quadruped_Robot

[6] Link to GitHub repository containing the code for simulation of calculations done in this project: https://github.com/ToyasDhake/Modelling_Project

Appendix

AgarwalRaghav.zip file contains a pycharm project which has implementation of simulation performed for the modelling of UMD SpotMini.

PyCharm project zip file contains:

1. LeggedRobot.py
2. motion_capture_data.py
3. movements.py
4. quadrupedPoseInterpolator.py

LeggedRobot.Py is the main file, it calls other files during compilation and execution. We can run LeggedRobot.py file to see the simulation performed. Details of the dependencies are mentioned in README.md file or on GitHub repository.

Project can also be found on GitHub repository: https://github.com/ToyasDhake/Modelling_Project

Demo Video link: https://www.youtube.com/watch?v=5Cuo5MBtMlU&feature=youtu.be