ENPM662

Introduction to Robot Modeling

Term project on:

Quadruped robot: UMD Spot Mini

Under the Guidance of:

Prof. Chad Kessens

University of Maryland, College Park

Submitted by

Toyas Dhake
116507271

Raghav Agarwal
115078055

Abstract

This project is a term project for subject ENPM662: Introduction to Robot Modeling. Performed by Toyas Dhake (116507271) and Raghav Agarwal (115078055) under the guidance of Prof. Chad Kessens. Project is on modeling of UMD Spot Mini based on Boston Dynamics' Spot Mini which has four legs each with 3 degrees of freedom and a gripper with five degrees of freedom.

The main objective of the project is to perform forward and inverse kinematics on the legs and gripper of the robot so that it can be used to get the legs and gripper in required orientation. Also, to decide the gait of the robot to walk but mostly focus will be on trot gait. For walking in a straight line, the trajectory of legs of the robot is required to be fixed, which we used 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.

In the end, robot should be able to pick and drop the object, walk-in a straight line, climb and get down from an obstacle.

Table of Contents

1 Introduction 5
1.1 Motivation 6
1.2 Aim 6
2 Robot Description 8
2.1 Link parameters and constraints 8
2.2 Assumptions 9
3 Methodology 10
3.1 Robot Modeling 10
3.1.1 Forward Kinematics 10
3.1.2 Inverse Kinematics 17
3.1.3 Validation 19
3.2 Walking 21
3.3 Climbing 24
3.4 Pickup 24
4 Simulations and Results 25
5 Conclusion and Future scope 29
5.1 Conclusion 29
5.2 Future scope 29
6 Bibliography 30

List of Figures

Figure 1 Simulation Environment 7
Figure 2 Robot Spawned at origin 11
Figure 3 Base co-ordinate frames of legs and gripper 11
Figure 4 Co-ordinate frames for each joint (leg) 13
Figure 5 Co-ordinate frame for each joint (gripper arm) 15
Figure 6 Inverse Kinematics (leg) 17
Figure 7 Inverse Kinematics (gripper arm) 18
Figure 8 Feet at (0, 0, 0) 20
Figure 9 Gripper at (0.5, 0.5, 0.5) 21
Figure 10 Support polygon for different situations 22
Figure 11 Trajectory of feet 23
Figure 12 Grip object 25
Figure 13 Stand up 25
Figure 14 Reach obstacle 26
Figure 15 Climb obstacle 26
Figure 16 Walk till end 27
Figure 17 Get down from obstacle 27
Figure 18 Drop object 28

Chapter 1

Introduction

Now-a-days robots have become an integral part of human life. We use robots for all sorts of different scenarios from manufacturing in industries and research in labs to recreation as a pet. There are 2 possible categories of robots i.e. stationary robots and mobile robots. A stationary robot can be used to do some sort of repetitive task such as manufacturing in industry. But if we must make generalize robots to do multitude of tasks then we must switch to mobile robots. Again, mobile robots can be classified into two categories i.e. wheeled robot and legged robot.

Wheeled robots are easier to implement and very efficient energy-wise to move from one point to another. But they have a huge limitation that they need properly constructed pathways. Mobility increases the workspace of a robot, but the limit of proper pathway limits the workspace. Moreover, if the robot is reliant on odometry to navigate and the wheel slips it can cause a huge problem. Legged robots don't have these limitations. Even though then have lower efficiency the benefits out ways the limitations in a lot of scenarios.

We need to ask what is engineering? Engineering is an iterative process that is used to optimize different activities we perform. We are doing this since last few centuries but there is one process occurring for few billion years optimizing things and creating marvels, Evolution. Wheels did not evolve in nature. No animals rolling around in nature with wheels. Right now, our legs are not as agile and dexterous as examples in nature say a cat, who can turn in midair to always land on feet while falling. But eventually, we will reach that point.

1.1 Motivation

Natural disasters occur all the time. We cannot do anything about disasters like earthquakes, floods and storms. All we can do is quickly overcome it reestablish our infrastructure that was damaged.

On January 12th, 2010 an earthquake of magnitude 7.0 hit just outside Port-au-Haiti, leveling a large part of the country. It was clear that even the earthquake lasted only 30 second the humanitarian disaster would last for years to come. Haiti had a bad economic situation it was the poorest country in the western hemisphere when the earthquake hit. Due to the scale of the destruction as soon as the word of disaster reach the world, focus of world's every major disaster response organization shifted to Haiti. But the issue was reaching needy people. Even though help arrived from neighboring countries, its logistics were difficult. Water and food are not useful if it does not reach the ones requiring it. Search and rescue had to be performed. These activities can be automated and thus, a smaller number of disaster relief people can reach more numbers of victims.

Such a robot would be more useful during disasters like Fukushima on March 11th, 2011, where humans simply cannot go. Due to the destruction all around wheeled robot cannot guarantee access to all the locations within disaster-affected zone, while the legged robot can either just walk over obstacles or jump over it.

1.2 Aim

We aim to designed to model a robot called UMD Spot Mini, modeled after Boston dynamics' Spot Mini. And to perform forward and inverse kinematics for each leg and gripper so that it can perform simple tasks like walking around, picking things up and crossing obstacles. Also use trot gait used to walk and keep robot stable by optimizing the motion of its leg.

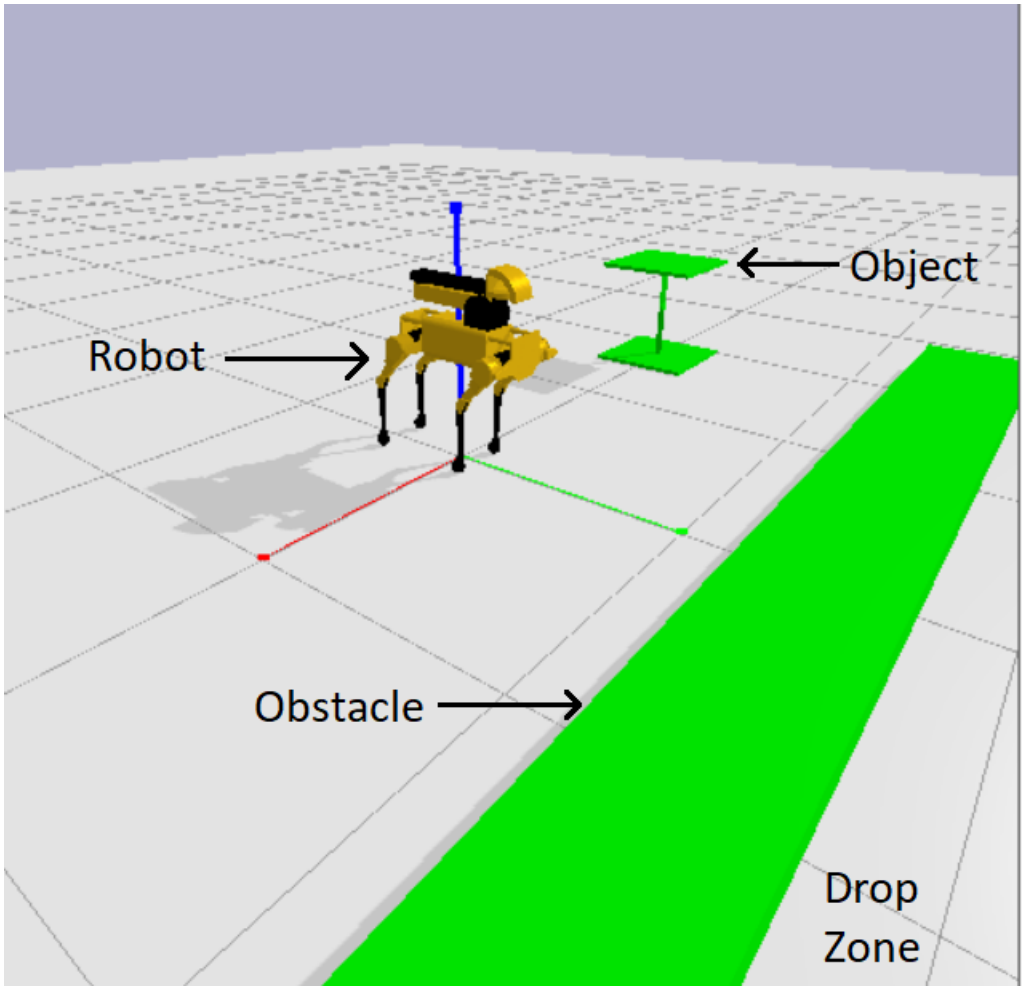

Figure 1 Simulation Environment

Figure shows the simulation environment of the robot. There is an object placed just by side the robot. It grabs the object walks towards the drop zone crosses the obstacle and drops the object in the drop zone.

Chapter 2

Robot Description

UMD Spot Mini has 4 legs each with 3 degrees of freedom. And a gripper on top of it with 5 degrees of freedom.

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

2.2 Assumptions

1. All links are rigid.
2. All joints are ideal with no friction.
3. Motors have infinite torque.
4. There is no backlash in motors.
5. There is infinite friction between robot's legs and ground.
6. There is infinite friction between robot's gripper and object.
7. The robot is moving quasi-statically.
8. Obstacle in the environment are static i.e. they have infinite mass and cannot be moved by robot.

Chapter 3

Methodology

Here is a brief description of the plan of implementing the project. The project was divided into milestones, as follows:

Task 1: Forward Kinematics and Inverse Kinematics for legs and gripper.

Task 2: Use inverse kinematics get to the object and grip it.

Task 3: Reset to original position.

Task 4: Start walking till obstacle is reached.

Task 5: Climb the obstacle.

Task 6: Walk Again till end.

Task 7: Get down the obstacle.

Task 8: Drop the object.

3.1 Robot Modeling

The modeling starts with transformation from the center of mass to the base of each leg and the base of gripper, then forward kinematics of all the legs and gripper. Next step is to extend this to inverse kinematics and get equations that can be used to get the desired orientation and position of the robot.

3.1.1 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.

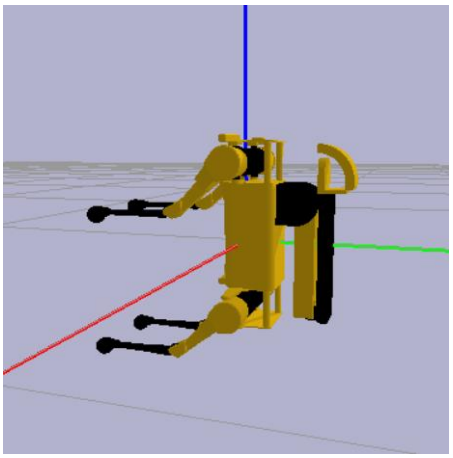

Figure 2 Robot Spawned at origin

The figure shows robot spawned with center of mass at origin. Basically, it shows the base coordinate frame.

Now we must transform from there to each leg.

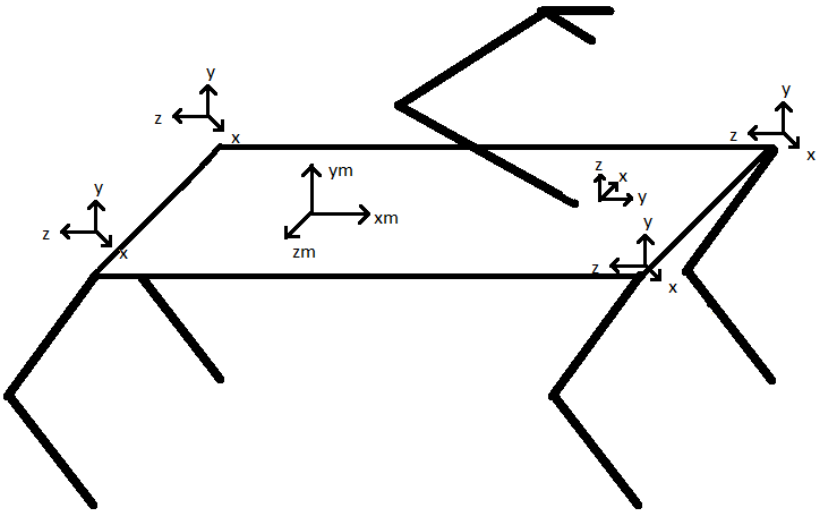

Figure 3 Base co-ordinate frames of legs and gripper

This transformation can be given by: -

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

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

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

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

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

Where T_m is the transformation applied to the base frame of the robot due to the robot's motion.

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

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

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

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

Forward Kinematics for Leg:

Now for each leg, the forward kinematics can be derived from following DH Table.

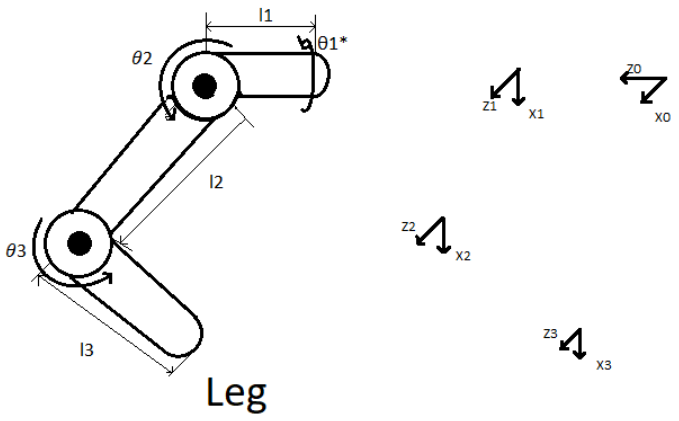

Figure 4 Co-ordinate frames for each joint (leg)

The figure shows coordinate frames chosen for the leg to calculate DH table. These frames follow the rules set for DH table. Z axis is along the actuator and x_{i+1} axis is mutually perpendicular to the z_i and z_{i+1} axis.

Note: Leg is not in zero position in the figure.

DH Table: -

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

Transformation matrix computed from this table are: -

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

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

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

$$T_3^0 = \begin{bmatrix} \cos(\theta_2+\theta_3)\, sin\theta_1 & -\sin(\theta_2+\theta_3)\, sin\theta_1 & cos\theta_1 & sin\theta_1[l_3\cos(\theta_2+\theta_3)+l_2 cos\theta_2] \\ -\cos(\theta_2+\theta_3)\, cos\theta_1 & \sin(\theta_2+\theta_3)\, cos\theta_1 & sin\theta_1 & -cos\theta_1[l_3\cos(\theta_2+\theta_3)+l_2 cos\theta_2] \\ -\sin(\theta_2+\theta_3) & -\cos(\theta_2+\theta_3) & 0 & l_1 - l_3\sin(\theta_2+\theta_3) - l_2 sin\theta_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

All the legs are identical hence we can use the computed forward kinematics as follows

For getting the coordinate of the end effector of a leg we can do 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- $T_{left\ front} * T_3^0$

Forward Kinematics for gripper arm:

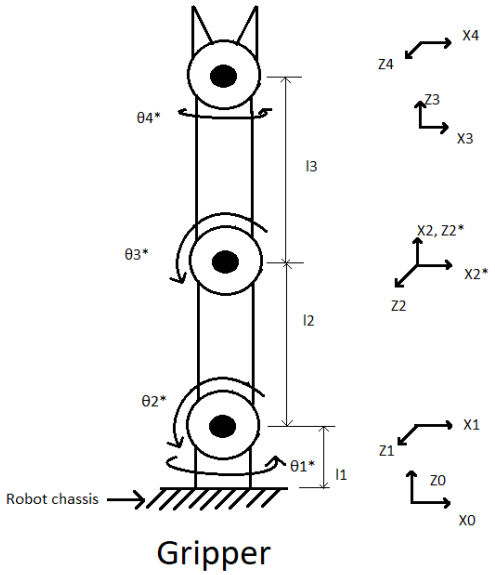

Figure 5 Co-ordinate frame for each joint (gripper arm)

Figure shows coordinate selection for the gripper.

DH Table: -

Frames	θ	d	a	α
0-1	θ_1*	g_1	0	90°
1-2	θ_2*+90°	0	g_2	0
2-2*	θ_3*-90°	0	0	-90°
2*-3	0	g_3	0	0
3-4	θ_4*	0	0	90°

Transformation matrix computed from this DH table: -

$$A_1 = \begin{bmatrix} \cos(\theta_1) & 0 & \sin(\theta_1) & 0 \\ sin(\theta_1) & 0 & -cos(\theta_1) & 0 \\ 0 & 1 & 0 & g_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_2 = \begin{bmatrix} -\sin(\theta_2) & -\cos(\theta_2) & 0 & -g_2\sin(\theta_2) \\ cos(\theta_2) & -\sin(\theta_2) & 0 & g_2\cos(\theta_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_3 = \begin{bmatrix} \sin(\theta_3) & 0 & \cos(\theta_3) & g_3\cos(\theta_3) \\ -cos(\theta_3) & 0 & \sin(\theta_3) & g_3\sin(\theta_3) \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

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

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

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}	$-cos\theta_1[(g_3)\sin(\theta_2 + \theta_3) + g_2 sin\theta_2]$
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}	$-sin\theta_1[(g_3)\sin(\theta_2+\theta_3)+g_2 sin\theta_2]$
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}	$g_1+(g_3)\cos(\theta_2+\theta_3)+g_2 cos\theta_2$
m_{41}	0
m_{42}	0
m_{43}	0
m_{44}	1

3.1.2 Inverse Kinematics

We need to find inverse kinematics for both, legs and the gripper.

Initially for legs, we will be using geometric method.

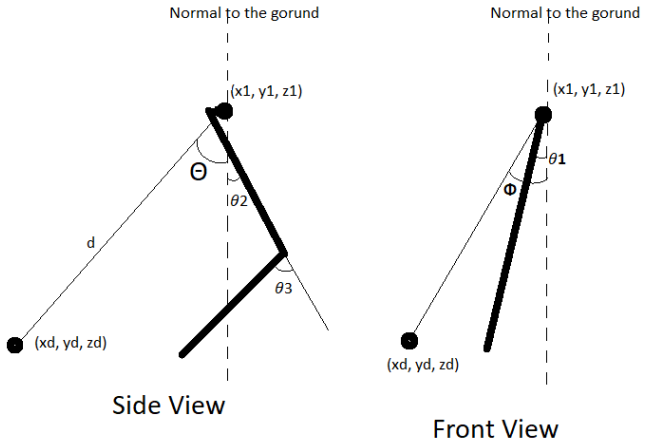

Figure 6 Inverse Kinematics (leg)

The above figure shows side view and front view of the leg. In the figure d is the absolute distance between base of the leg and desired coordinate.

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

Φ is the angle made by the line between leg base and desired point with normal to the ground in Y-Z plane. Hence, we need to set the θ_1 angle to it.

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

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

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

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

$$\theta_3 = 180\ - acos\left(\frac{l_2^2 + l_3^2 - d^2}{2l_2l_3}\right)$$

These same equations will apply for all the legs as they are exactly same.

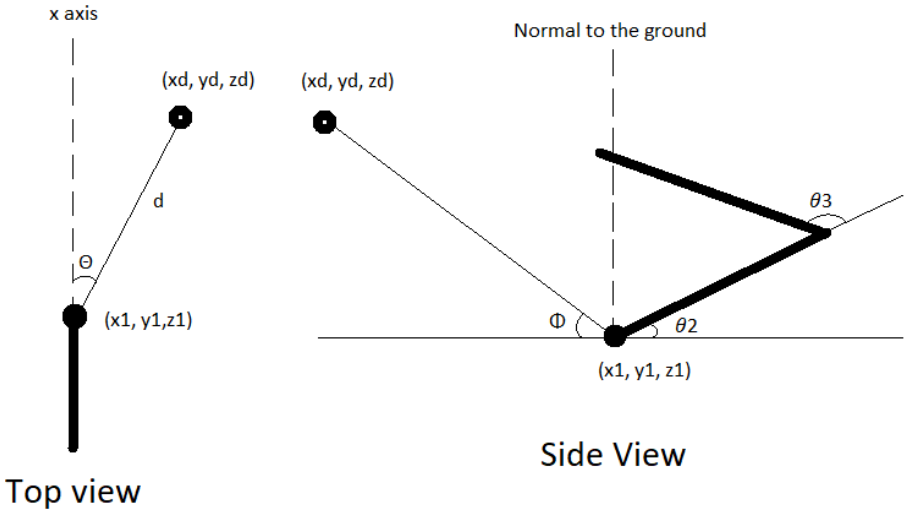

Figure 7 Inverse Kinematics (gripper arm)

The above figure shows top view and side view of the arm. In the figure d is the absolute distance between base of the leg and desired coordinate.

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

Θ is the angle made by the line between base of gripper arm to the desired point with X axis in X-Z plane. Hence, we can directly set θ_1 to that value.

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

Φ is the angle made by the line between gripper arm base to desired point with parallel to ground. Hence, to get θ_2 we need to subtract Φ plus 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 know all sides of triangle. θ_3 is an external angle hence we need to subtract calculated angle from 180°.

$$\theta_3 = acos\left(\frac{l_2^2 + l_3^2 - d^2}{2l_2l_3}\right)$$

θ_4 of the gripper arm is just for orientation. It can be set to any value based on orientation of object as it does not have any rotation constrains.

θ_5 is just the gripping action. 0° means the gripper is holding and 180° means the gripper is open.

3.1.3 Validation

Validation will have 2 cases: -

1. Validation of inverse kinematics for legs we will try to: -

If we spawn the robot to stand at origin, the center of mass of the robot is at (0, 0.5, 0). The base of rear left leg will be at (-0.2, 0.5, -0.08). We will compute θ_1, θ_2 and θ_3 for feet to be at (0,0,0).

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

$$\theta_1 = 9.0902°$$

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

$$d = 0.544$$

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

$$\theta_2 = -31.086$$

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

$$\theta_3 = 17.013°$$

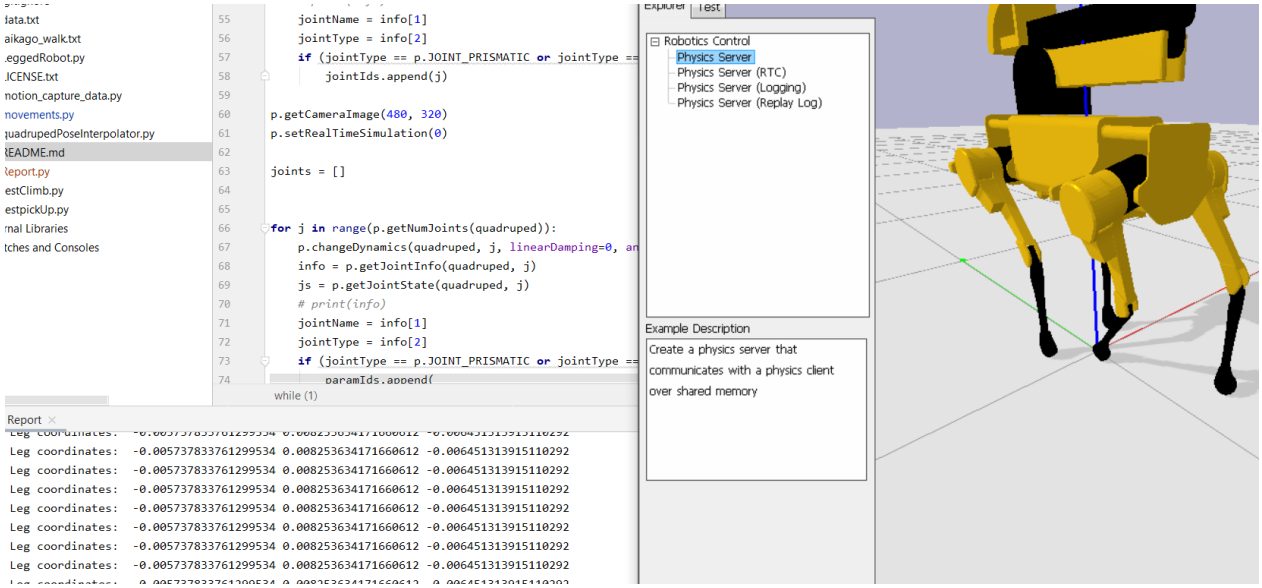

Figure 8 Feet at (0, 0, 0)

As it can be seen leg is roughly at (0,0,0). The coordinates from the simulator for feet are displayed in bottom left corner. They are roughly at (0,0,0).

2. Validation of inverse kinematics for gripper: -

Like the leg we will take a destination point for the gripper. Let destination point be (0.5,0.5,0.5). Base of the gripper will be at x + x_g i.e. (0.12, 0.5, 0). We will compute θ_1, θ_2 and θ_3 for this position.

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

$$\theta_1 = 52.765°$$

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

$$d = 0.628$$

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

$$\theta_2 = 148.861°$$

$$\theta_3 = acos\left(\frac{0.35^2 + 0.375^2 - 0.628^2}{2 \times 0.35 \times 0.375}\right)$$

$$\theta_3 = 120°$$

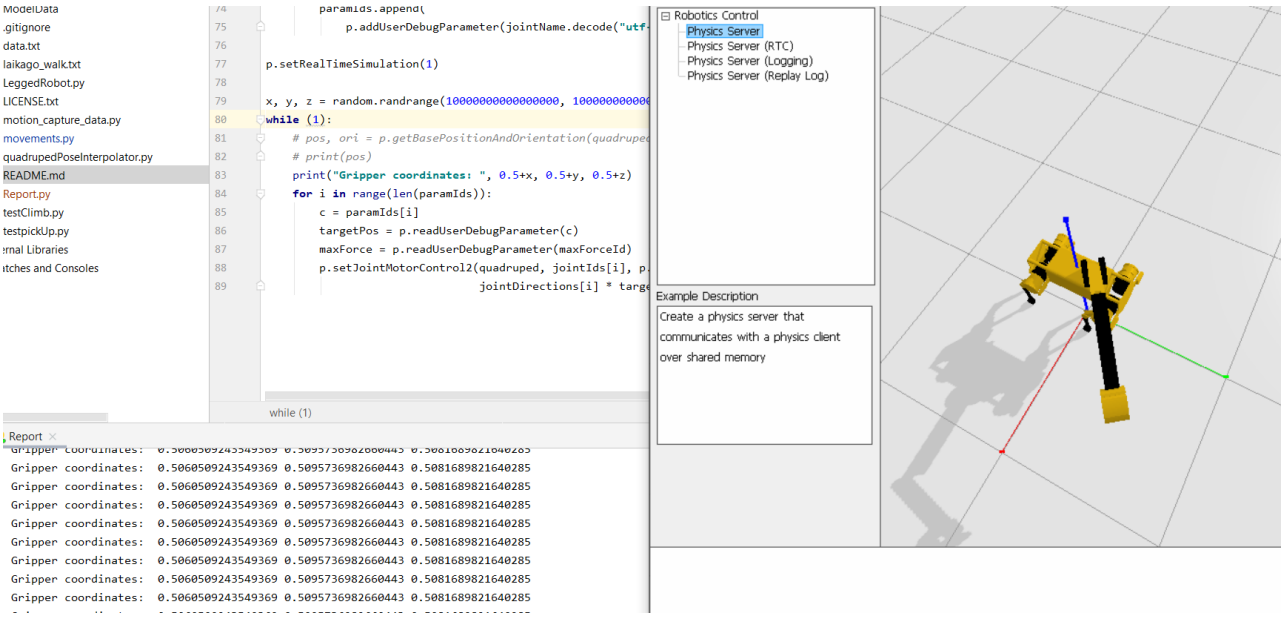

Figure 9 Gripper at (0.5, 0.5, 0.5)

As it can be seen gripper is roughly at (0.5,0.5,0.5). The coordinates from the simulator for gripper are displayed in bottom left corner. They are roughly at (0.5,0.5,0.5).

3.2 Walking

Walking can be achieved in multiple different ways namely:

1. Walk-

   In walking gait only one leg of the robot is off the ground. Because of this it is the most stable way of motion as the support polygon is always a triangle. And it is easy to keep center of mass inside the support polygon. But it has slowest motion.

2. Trot-

   In tort gait 2 diagonally opposite legs are in the air at a time. Due to this the support polygon is a line. It is not easy to balance the center of mass on the line, but this results in faster motion.

3. Gallop-

   In gallop all the legs touch the ground one at a time and rest of the time robot is in air. Robot uses the time during which the legs touch the ground to launch itself into the air. This results in fastest possible motion. Gallop needs very optimized controller to implement. As controls is not in the scope of this project, we will not investigate gallop.

Walk Vs Trot

As mentioned above walk is more stable when compared with trot. But it is too slow. Initially we tried walk by picking one leg at a time. But robot was not walking in straight line.

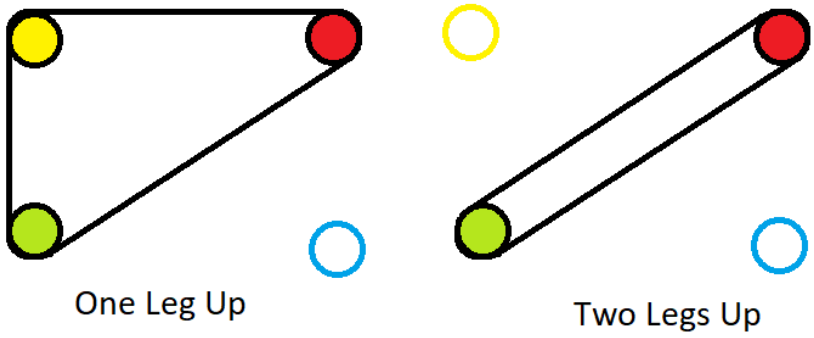

Figure 10 Support polygon for different situations

Walking mechanism:

If we look at the robot without gripper, it is symmetric hence the center of mass is roughly in at the geometric center. But after adding the gripper. The center of mass shifter significantly forward. We used this to our advantage.

1. Lift on of the front leg slightly.
2. Wait for some time so the robot will fall forward as center of mass is forward.

3. Diagonally opposite leg will be in air now. Based on forward kinematics and previous info about the location of other 2 legs we can calculate location where the leg in air is supposed to be.
4. Now extend the front leg to original position.
5. Repeat the same with other front leg.

Using this the robot moved forward successfully but it starts turning right after few steps. This maybe because the center of mass is not exactly on center axis maybe there is an offset.

Trot Mechanism:

In trot 2 legs are in air at a time, this results in instability. But we were waiting for robot to fall on front leg. Instead in trot we moved the leg quickly so the robot will not have time to fall. We used 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 should move in perfect D shape.

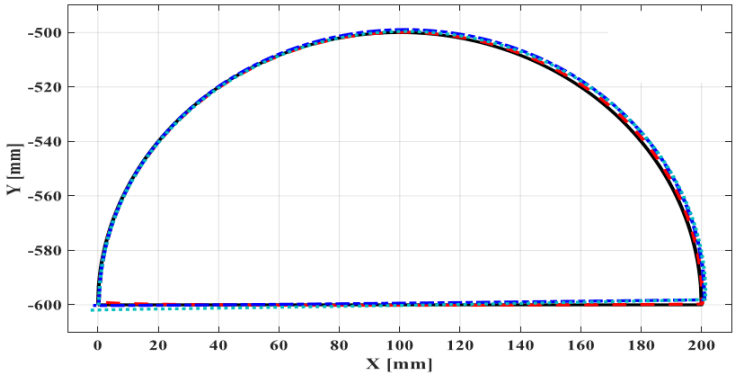

Figure 11 Trajectory of feet

Scale of the robot leg in the simulation used was different hence values on axis does not make sense but the results were scalable, and our robot successfully followed a D shape in every cycle. Adjacent legs are always at opposite point on the curve. So, when one leg is down on ground and pushing behind the other one is in air moving forward. This cycle was set to be repeated in 500 milliseconds. Which means robot takes 2 steps per second. This cycle time resulted is stable walk. Change its speed would result in instability.

3.3 Climbing

For climbing we took advantage of same thing. We quickly placed both legs of the robot on the platform. Then used rear legs to push the robot on the platform.

3.4 Pickup

Picking action was easy as contact modeling is not in scope of this project. The object is made of a dumbbell shape so that it won't fall of the gripper. Using the inverse kinematics derived above the arm can reach the object and grab it.

Chapter 4

Simulations and Results

Based on the equations developed above, the robot is capable of performing all the tasks which were initially set. Following are screenshots of the simulation at different time instances. After completion of specific task.

Task 2: Use inverse kinematics get to the object and grip it.

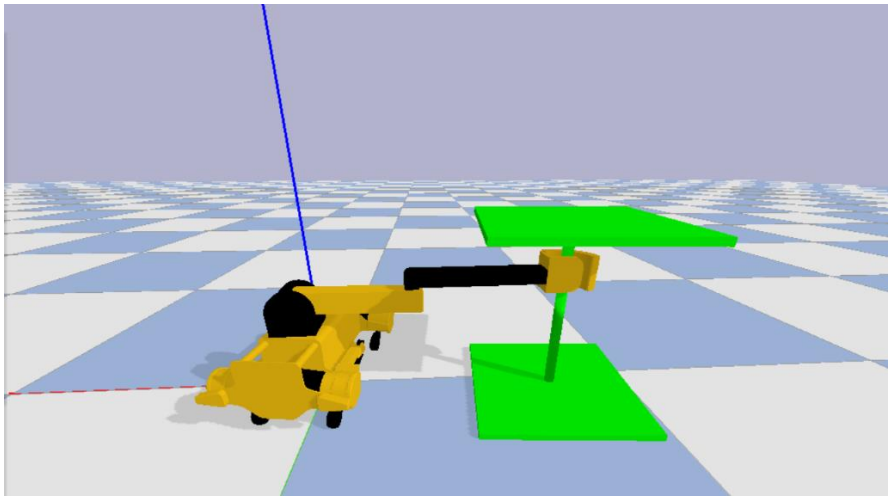

Figure 12 Grip object

Task 3: Reset to original position.

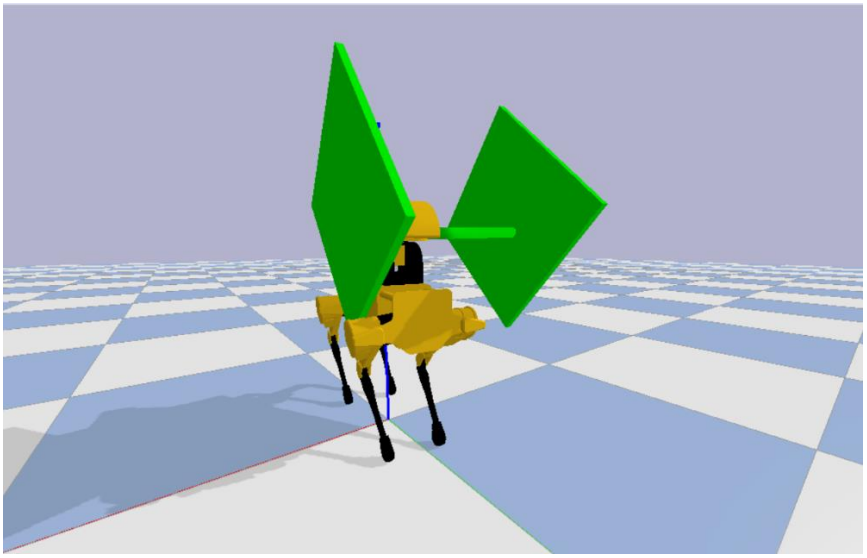

Figure 13 Stand up

Task 4: Start walking till obstacle is reached.

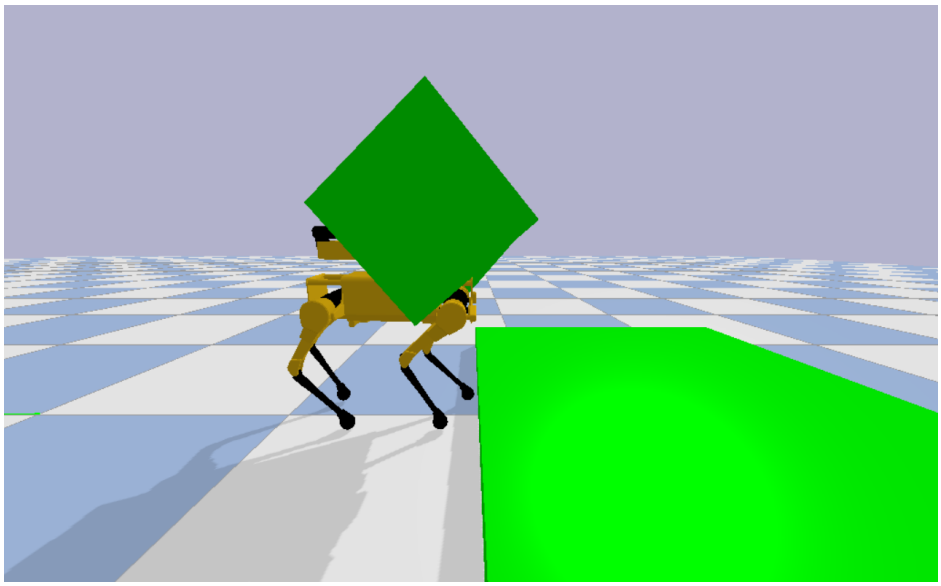

Figure 14 Reach obstacle

Task 5: Climb the obstacle.

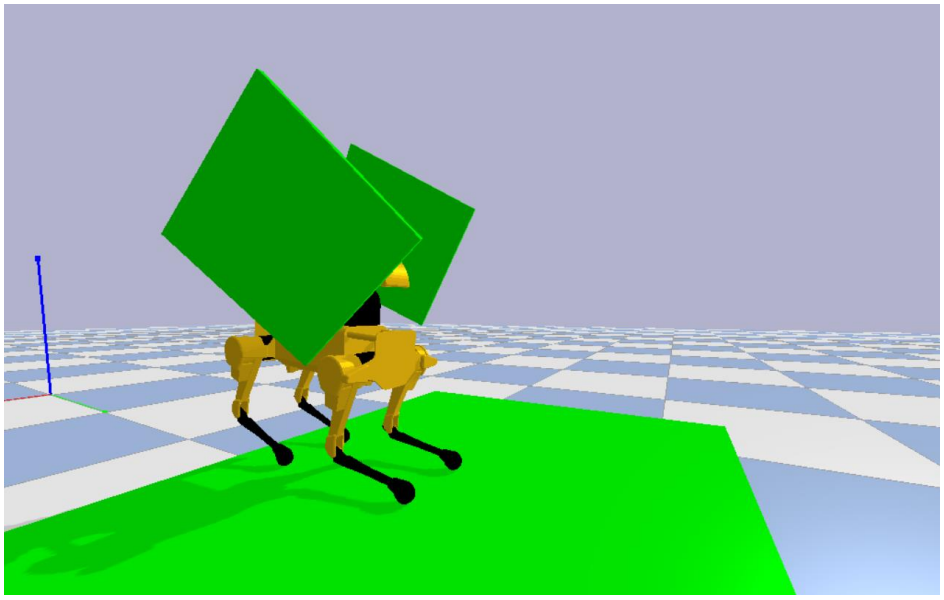

Figure 15 Climb obstacle

Task 6: Walk Again till end.

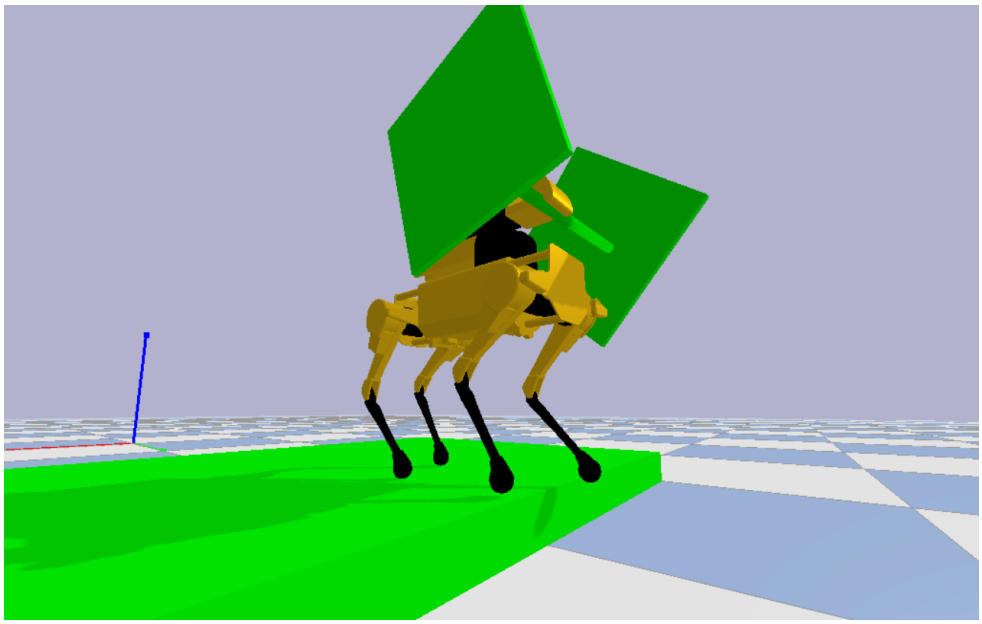

Figure 16 Walk till end

Task 7: Get down the obstacle.

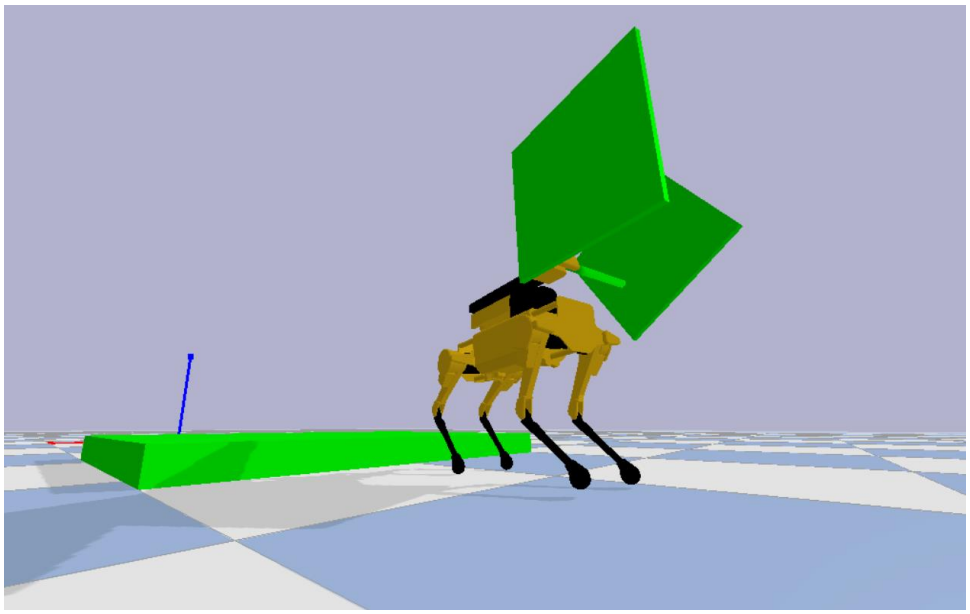

Figure 17 Get down from obstacle

Task 8: Drop the object.

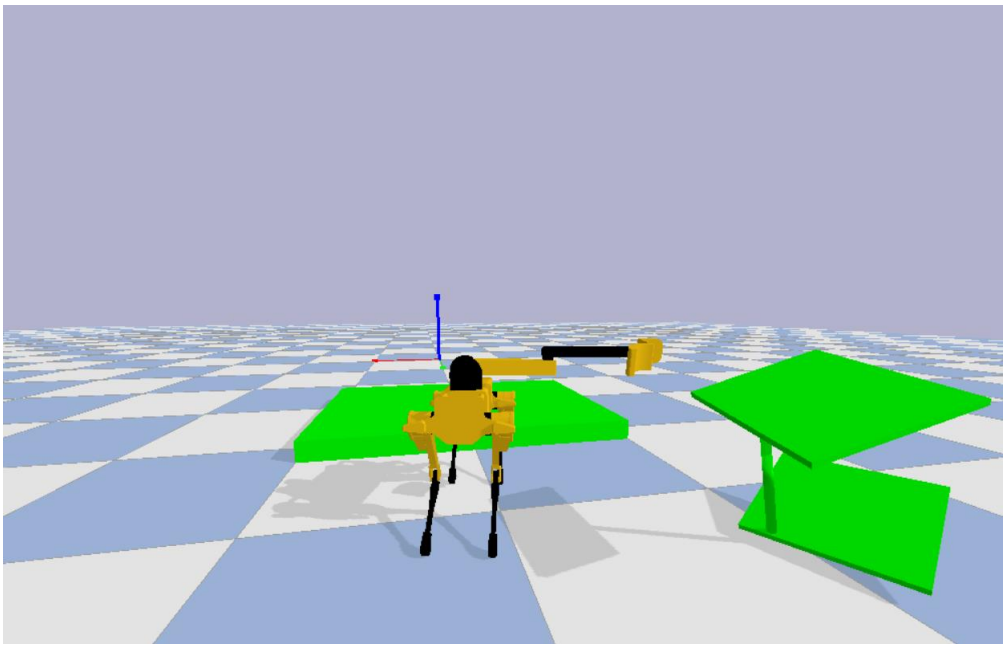

Figure 18 Drop object

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

Chapter 5

Conclusion and Future scope

5.1 Conclusion

The aim of this project was to compute forward and inverse kinematics for the robot and take an object from point A to point B, which was done in Pybullet using Unitree's Laikago as base model, modified in SoildWorks to add gripper. There is a lot of room for improvement. Right now, the speed of robot is locked. If we increase or decrease speed of motion of legs the system becomes unstable.

There is problem 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.

Finally, I would like to thank Prof. Chad Kessens for allowing me to work on this project and giving me insights without which project might not have got completed and TAs for helping and supporting me throughout the course.

5.2 Future scope

There are unlimited possibilities that can be implemented on UMD Spot Mini. Different sensors can be used to create a sensor fusion for the robot so that it can sense its environment and act dynamically. If the falls on its back a sequence can be developed so that it can get back up on its feet. Different walking gaits can be implemented such as walking and galloping which may prove useful in certain scenarios. Lastly, the main advantage of legs over wheels can be implemented that is jumping.

Bibliography

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

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

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

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

Appendix

Final_Project_Toyas_Dhake.zip file contains a pycharm project which has implementation of simulation this project.

PyCharm project contains multiple files: -

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

out of which LeggedRobot.Py is the main file, it uses other files to execute. Run LeggedRobot.py to see demonstration. 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