Indian Institute of Technology Patna

ME335

Bio-Inspired Robotics

November 18

Final Report 2019

Sub – Objective:
Design concept and CAD model, containing basic mechanism, calculations and motor selection.

Lizard Inspired Four Legged Bot

Group – 01:
1701ME04: Aman Deep
1701ME12: Bansal Kumar
1701ME14: Chandan Bhat
1701ME29: Prateek Gupta
1701CS26: Manavjeet Gupta

Project Objective

To design a **four-legged lizard-inspired robot** (no wheels) with following functions and properties:

- It can travel on a straight line 25 times its body's largest dimension in two minutes or less.
- Maximum body length should be less than 30 cm.
- Off-the-shelf motors must be used.
- The batteries and the controllers must be embedded in the robot structure and robot operation should be untethered.
- FDM is to be used for making structures.
- Programming of robot locomotion is required.

Introduction

Bio-Inspiration

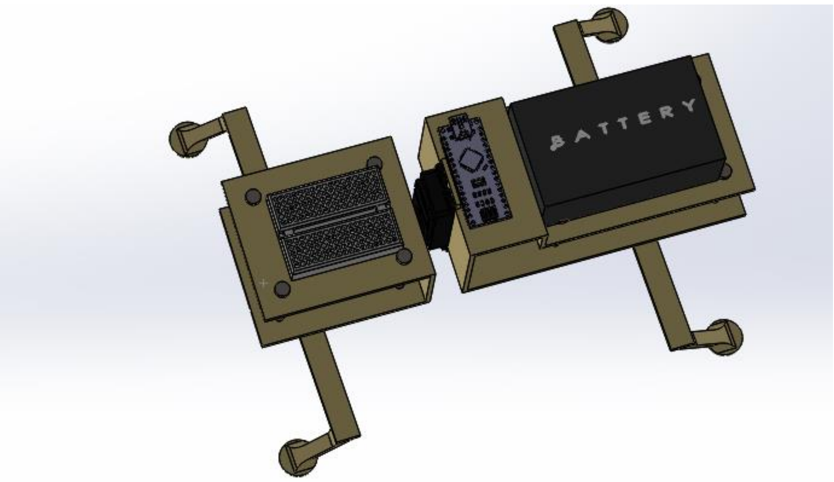
Most lizards are quadruped and move using gaits with alternating movement of the right and left limbs with substantial body bending. The muscles of its body help it in achieving a perfectly synchronized motion. The centre of gravity of the lizard is very low giving it good static as well as dynamic stability.

Legged Robots

Legged robots are a type of mobile robot, which use articulated limbs, such as leg mechanisms, to provide locomotion. They are more versatile than wheeled robots and can traverse many different terrains, though these advantages require increased complexity and power consumption.

Design Concept

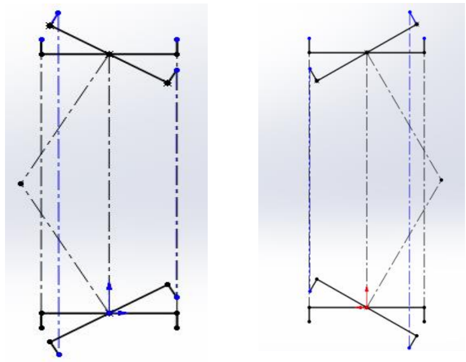
CAD Model



Description

The body is mainly divided in two parts, each having a set of legs. Each set moves corresponding to one motor only. The two parts are joined together in middle which includes a third motor which rotates the two parts lifting one side of the body in both the parts. It also includes space for battery, Arduino Nano, mini breadboard and wires.

Walking Mechanism



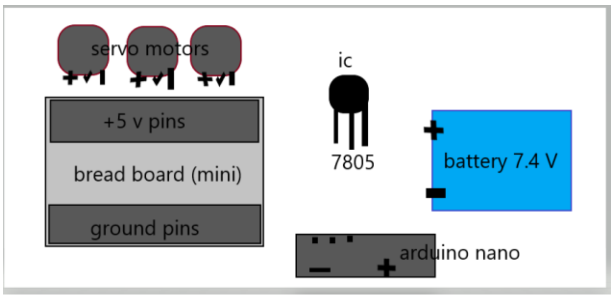
- 4 Leg-Gait Walking Mechanism.
- Front legs point forward while rear leg points backward.
- 1 motor actuates front legs and the other rear ones.
- A third motor is used to twist the body from middle

Leg Configurations

Four legs are constructed to provide ample ground clearance. There is an extension from the body away which bends downward. This bend is chosen as to be front bend in front legs and reverse bend in rear legs. This configuration is direct concept from lizard's own body. This configuration is also required based on above pictures.

Also hemispherical end to the leg is given at contact points to have desired normal reactions and grip is attached to get sufficient coefficient of friction.

Electronics



- Battery Li-Po 7.4 V
- IC 7805 (for voltage regulation)
- Arduino-Nano
- Bluetooth Module (HC05)
- Mini Bread-Board
- Servo-Motor 10.6 kg-cm torque

Calculation of Torque for Servo-Motor

Length of One End of Leg from Mid-Plane = 70 mm

Weight = 0.6 kg and Factor of Safety = 2

Hence Required Moment = 7 cm x 2 x 0.6 kg = **8.4 kg cm.**

Bot Characteristics

Duty Factor

Total Time = 1.2 s with speed of motor 60° per 0.16 s.

For Central Motor:

Operating Time = (15/60)*0.16s = 0.04s

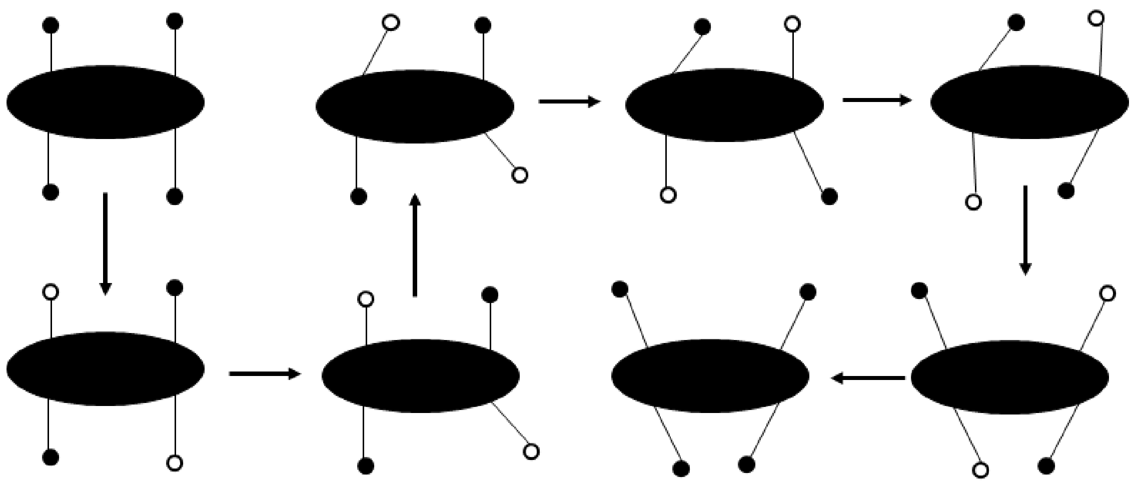
Duty Factor = 0.033

Front and Rear Motor:

Operating Time = (60/60)*0.16s = 0.16s

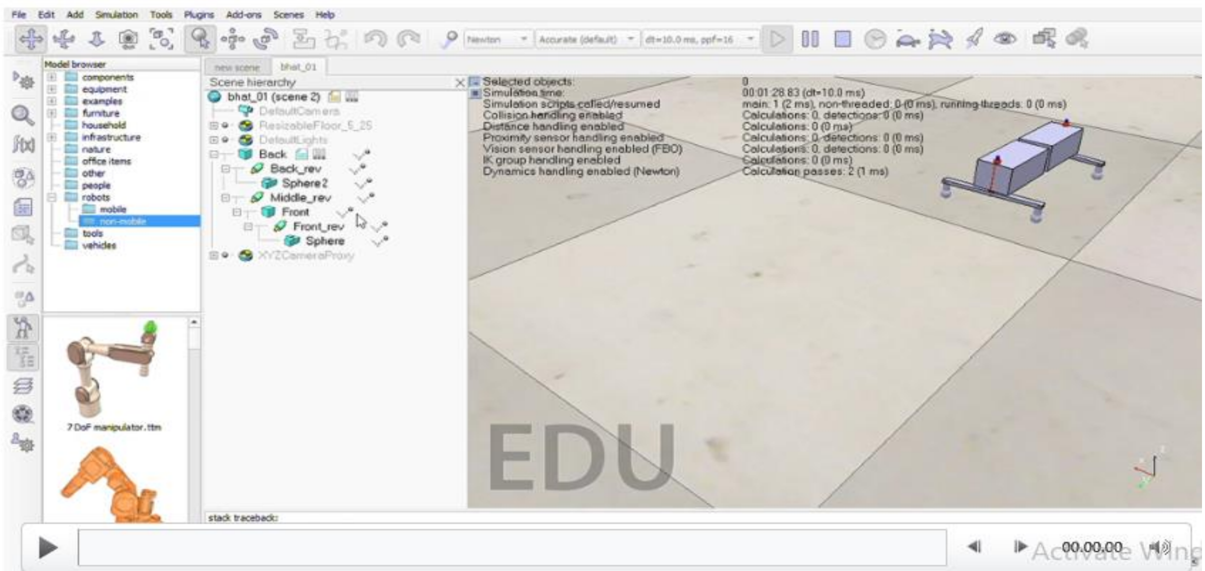
Duty Factor = 0.133

Gait Diagram



V-Rep Analysis

Video (Screenshot)

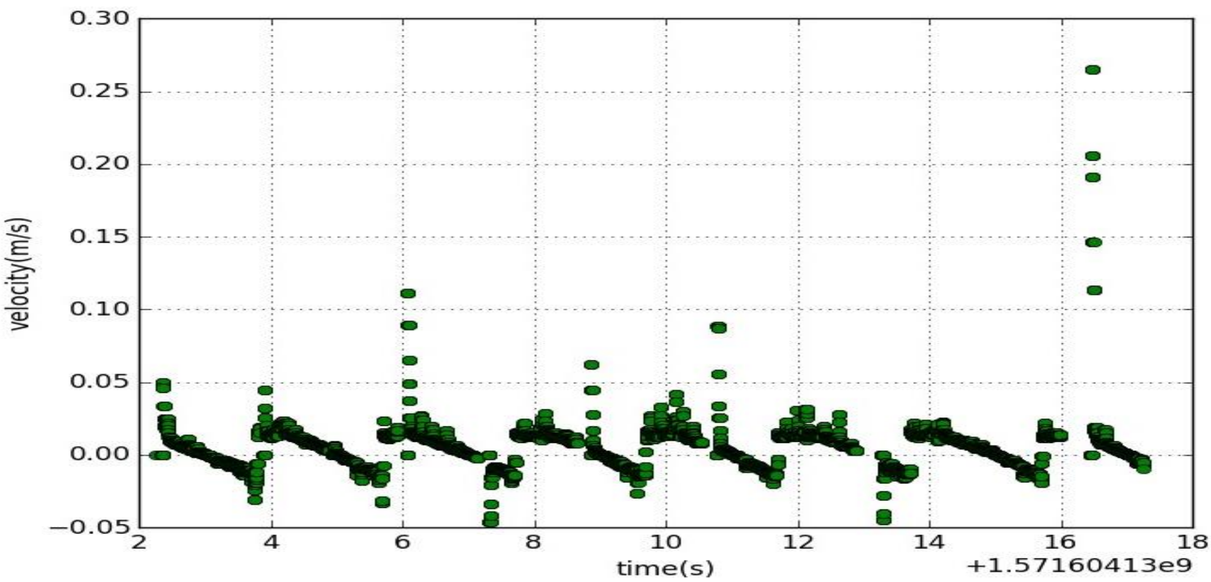


Code

```python


res,servo_front=vrep.simxGetObjectHandle(clientID,'Front_rev',vrep.simx_opmode_blocking)
res,servo_back=vrep.simxGetObjectHandle(clientID,'Back_rev',vrep.simx_opmode_blocking)
res,servo_middle=vrep.simxGetObjectHandle(clientID,'Middle_rev',vrep.simx_opmode_blockin
res,body_back=vrep.simxGetObjectHandle(clientID,'Back',vrep.simx_opmode_blocking)

print('dude')
t = time.time()
z= 0;
v= 0;
while (time.time()-t ) <= 15:

    res,linearVelocity,angularVelocity = vrep.simxGetObjectVelocity(clientID,body_back,v

    print('vy' , linearVelocity[1])

    z = z+1;
    v= v+ (linearVelocity[1]);

    plt.plot(time.time(),linearVelocity[1],'go--')
    plt.xlabel('time(s)')
    plt.ylabel('velocity(m/s)')
    plt.grid(True)

    plt.hold(True)


    tf,tr,tm= phase(time.time())
    vrep.simxSetJointTargetPosition(clientID,servo_front, tf,vrep.simx_opmode_oneshot)

    vrep.simxSetJointTargetPosition(clientID,servo_back, tr,vrep.simx_opmode_oneshot)
    vrep.simxSetJointTargetPosition(clientID,servo_middle, tm,vrep.simx_opmode_oneshot)


print(v/z);

```

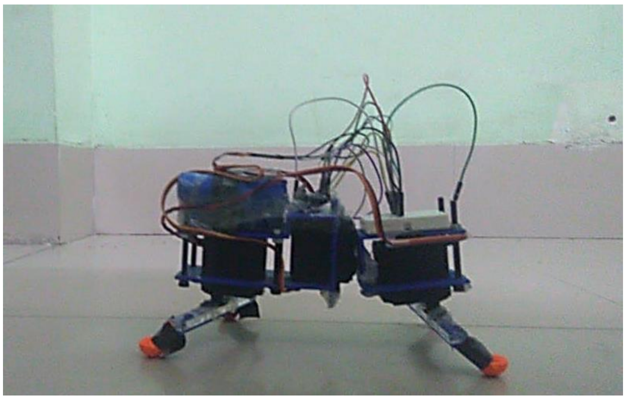
```
('vy', 0.030541419982910156)
('vy', 0.030541419982910156)
('vy', 0.030541419982910156)
('vy', 0.030541419982910156)
('vy', 0.030541419982910156)
('vy', 0.030541419982910156)
('vy', 0.030541419982910156)
('vy', 0.030541419982910156)
('vy', 0.030541419982910156)
('vy', 0.030541419982910156)
('vy', 0.027203559875488281)
('vy', 0.027203559875488281)
('vy', 0.027203559875488281)
('vy', 0.027203559875488281)
('vy', 0.027203559875488281)
('vy', 0.027203559875488281)
('vy', 0.027203559875488281)
('vy', 0.027203559875488281)
('vy', 0.027203559875488281)
('vy', 0.024402141571044922)
('vy', 0.024402141571044922)
('vy', 0.024402141571044922)
('vy', 0.024402141571044922)
('vy', 0.024402141571044922)
('vy', 0.024402141571044922)
('vy', 0.024402141571044922)
('vy', 0.024402141571044922)
('vy', 0.024402141571044922)
('vy', 0.024402141571044922)
('vy', 0.01760721206665039)
('vy', 0.01760721206665039)
('vy', 0.01760721206665039)
('vy', 0.007033348083496094)
('vy', 0.007033348083496094)
('vy', 0.007033348083496094)
('vy', 0.007033348083496094)
0.0136120650824
>>>
```

Velocity Graph



Manufacturing Process

1. Procured Parts
2. 3-D Printing of Chassis
3. Assembling Final Model
4. Finishing the Circuit
5. Testing Basic Code
6. Testing on Different Terrains
7. Improving Coefficient of Friction



Arduino Code

```
#include <Servo.h>

// create servo object to control a servo
Servo front;
Servo mid;
Servo rear;

int val=0;    // variable to give initial position command

void setup() {
  front.attach(9);
  mid.attach(6);
  rear.attach(3);
  // attaches the servo pins to the servo object
}

void loop() {
  //to set initial position
  while(val<1)
  {
   front.write(90);                  // sets the servo position according to the scaled value
   mid.write(90);
   rear.write(90);
   delay(200);                        // waits for the servo to get there
   val++;
  }


 //first half of cycle
   mid.write(85);
   delay(200);
   front.write(110);
   delay(200);
   rear.write(70);
   delay(200);

 //2nd half of the cycle
   mid.write(95);
   delay(200);
   front.write(70);
   delay(200);
   rear.write(110);
   delay(200);
 }
```

Performance in Different Terrains

S. No.	Property	Smooth (Tile) Surface	Tennis Mat	Mud Cover
1.	Screenshot			
2.	Coefficient of Friction	Low	High	Medium
3.	Distance Travelled (in 2 min)	~14 Body Lengths	~22 Body Lengths	~19 Body Lengths

"Best performance was obtained on the **mat** as it gave the best coefficient of friction."

Reason for Less Travel: Because the normal reaction is less on the legs (bot weight is less), hence the legs are not able to hold their positions and slip in backward direction hence decreasing the speed.

Errors in Manufacturing

1. Improper Alignment of Central Motor
2. Non-Rigidity due to Bending Factor in Structure and Legs due to weight.
3. Unequal Reactions on Support Points
4. Unequal Weight Balance (taken care but not exact)
5. Joining of Servos cannot be Perfect.
6. Imperfection at Joints

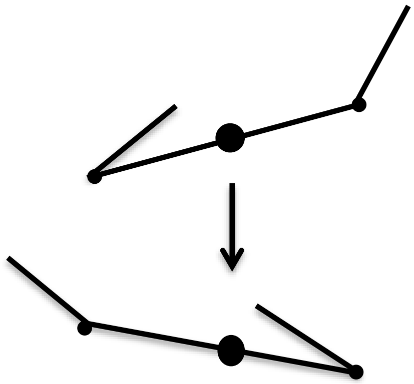
Scope of Improvement

1. Friction at the Legs can be Improved.
2. Phase can be Better Synchronized for Faster Travel.
3. Synchronization between Amplitude and Time Period.

4. Errors in Manufacturing can be Improved Upon

Further Enhancement in Design

1. There is also a movement in the extended arms of legs which we currently have fixed. This motion is demonstrated alongside.



2. The control of the bot is thought to be made more sophisticated by giving it the ability of moving sideways and backwards as well.

Cost Analysis

S. No.	Component	Price per Unit	Quantity	Net Price
1.	Arduino Nano	300	1	300/-
2.	Servo Motors	330	3	990/-
3.	Mini-Breadboard	50	1	50/-
4.	Battery (LiPo- 7.4V)	400	1	400/-
5.	Chassis	-	-	~200/-
6.	Jumper Wires	1	~20	20/-
7.	Miscellaneous	-	-	~150/-
	Total			**2110/-**

These prices are approx. Hence we can take as **2100** to be the cost of our entire bot.