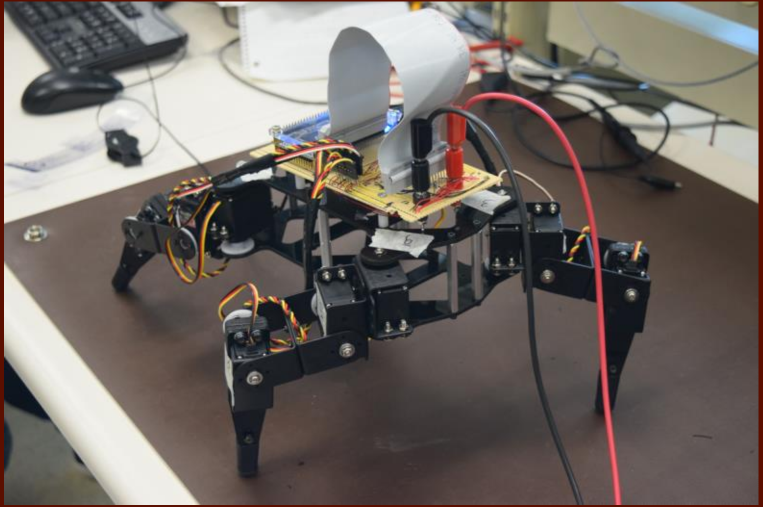
Smart Robotic Quadruped

Brittany Lamorie
Duncan Prance
Cody Otto



With Special Guest:
Bob the Robot

Motivation

- Interest in robotics and machine learning
- Walking is an interesting challenge
- ECE 449 Intelligent Systems

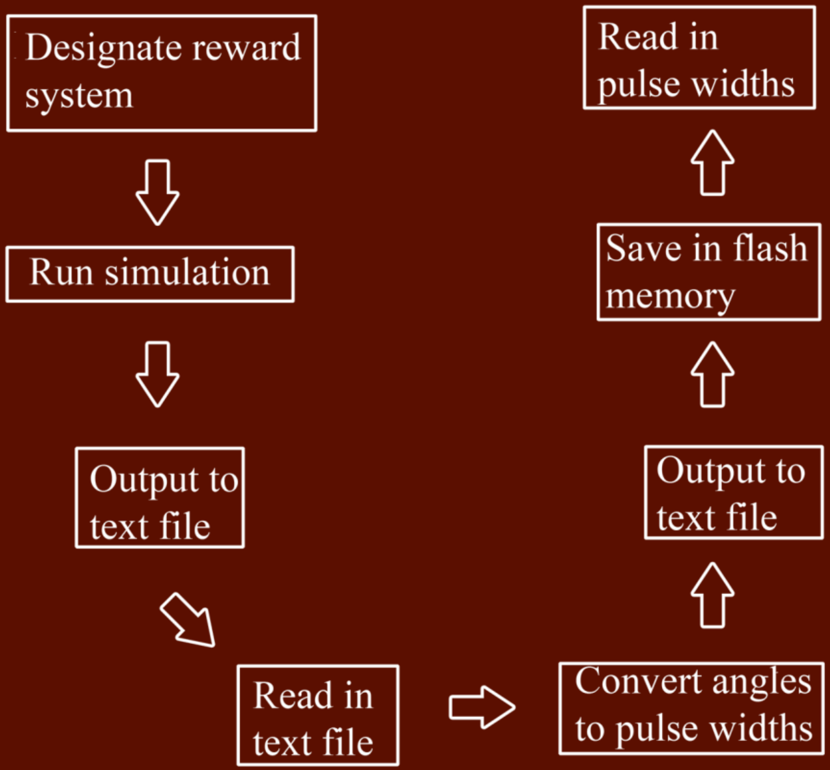
Goals

- Build working robot
- Create accurate simulation of physical robot
- Convert simulation output to pulse width signals
- Hard-code basic motions
- Learned walking motion
- On board adjustments

Overview

Design Plan

- Simulation is run; All learning done on PC
- Simulation outputs are converted to pulse width signals
- Pulse widths are saved in memory
- On board program reads pulse widths


Designate reward system

Run simulation

Output to text file

Read in text file

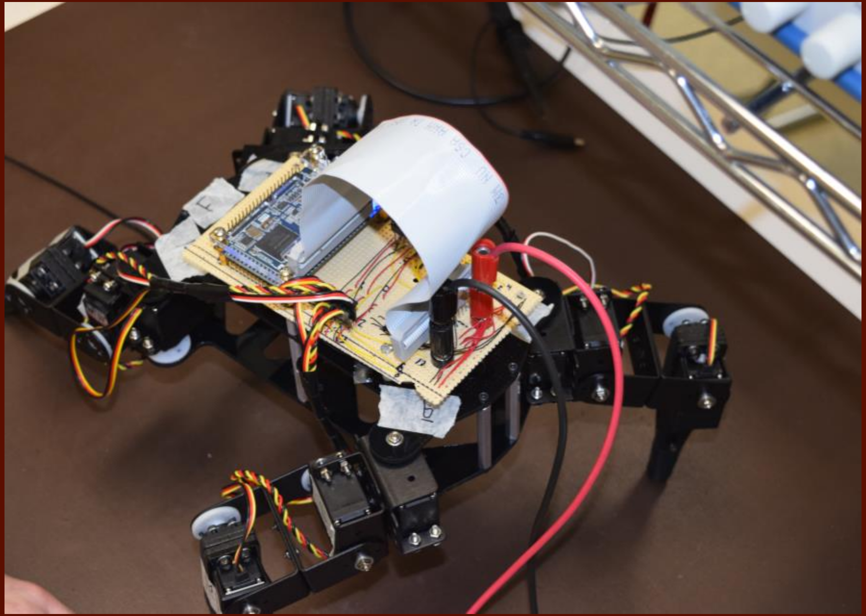
Convert angles to pulse widths

Output to text file

Save in flash memory

Read in pulse widths

Overview

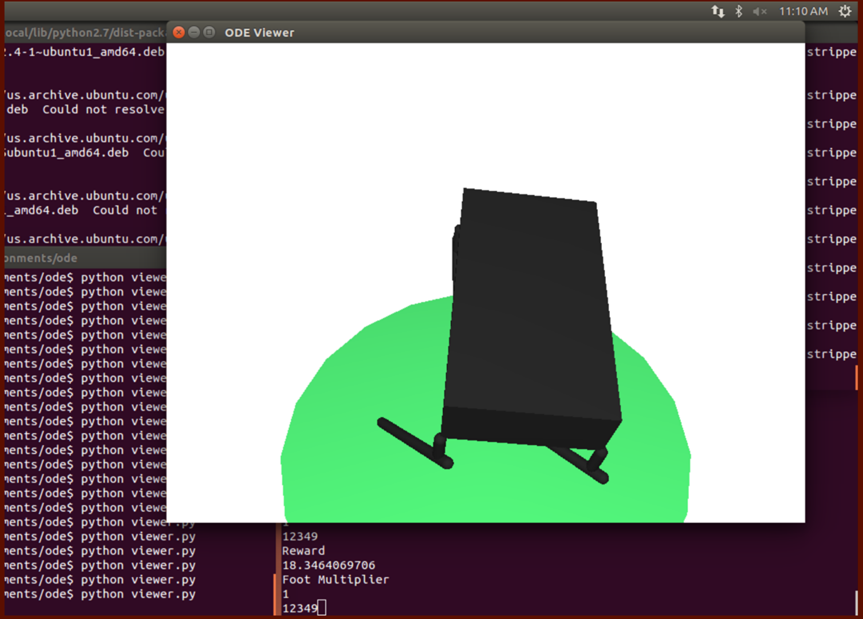


Current Functionality

- Hardcoded Modes
  - Walk
  - Dance
  - Weight Shift
  - Simulation
- Simulation output conversion
- Battery powered tether

Simulation

- Simulation used a combination of ODE and OpenGL for physics simulating, and PyBrain for the machine learning aspect
- The machine learning uses a reward based system, allowing for fine tuned control over learned motion
- The simulation is taught through a combination of neural networks and reinforcement learning



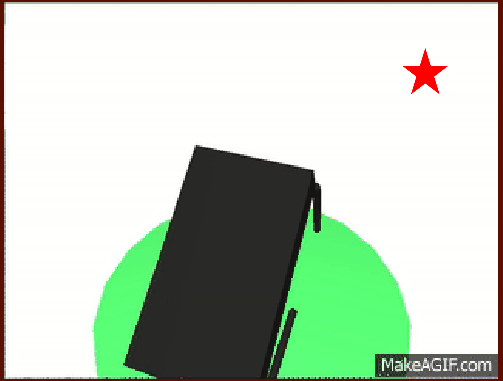
```
.4-1~ubuntu1_amd64.deb

/us.archive.ubuntu.com/
 deb  Could not resolve

/us.archive.ubuntu.com/
ubuntu1_amd64.deb  Cou


/us.archive.ubuntu.com/
_amd64.deb  Could not

/us.archive.ubuntu.com/
```

```
stripped
stripped
stripped
stripped
stripped
stripped
stripped
stripped
stripped
stripped
stripped
```

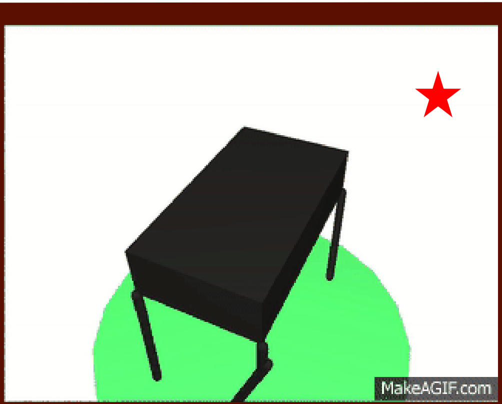
```
ments/ode$ python viewe
ments/ode$ python viewe
ments/ode$ python viewe
ments/ode$ python viewe
ments/ode$ python viewe
ments/ode$ python viewe
ments/ode$ python viewe
ments/ode$ python viewe
ments/ode$ python viewe
ments/ode$ python viewe
ments/ode$ python viewe
ments/ode$ python viewe
ments/ode$ python viewe
ments/ode$ python viewe
ments/ode$ python viewe
ments/ode$ python viewe
ments/ode$ python viewe
ments/ode$ python viewer.py
ments/ode$ python viewer.py
ments/ode$ python viewer.py
ments/ode$ python viewer.py
ments/ode$ python viewer.py
ments/ode$ python viewer.py
```

```
1
12349
Reward
18.3464069706
Foot Multiplier
1
12349
```

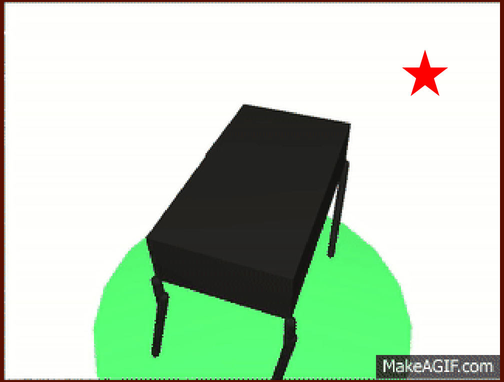
Simulation



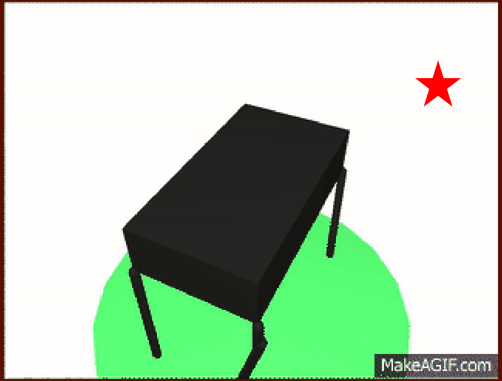
Trial 1



Trial 740



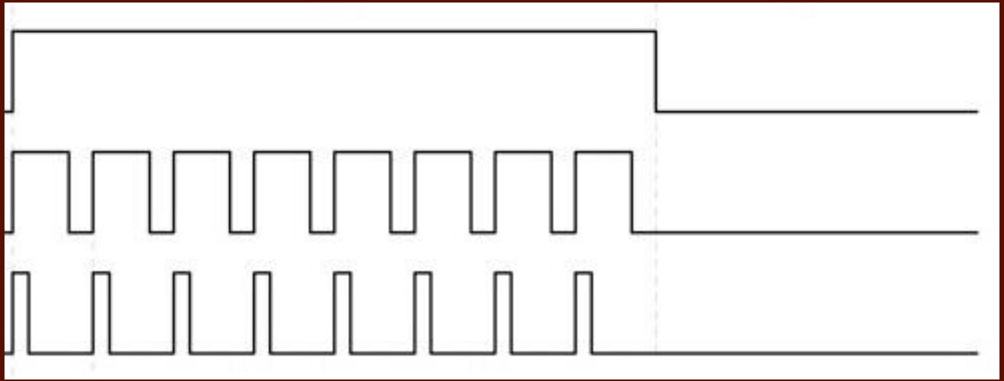
Trial 10500



Trial 20000

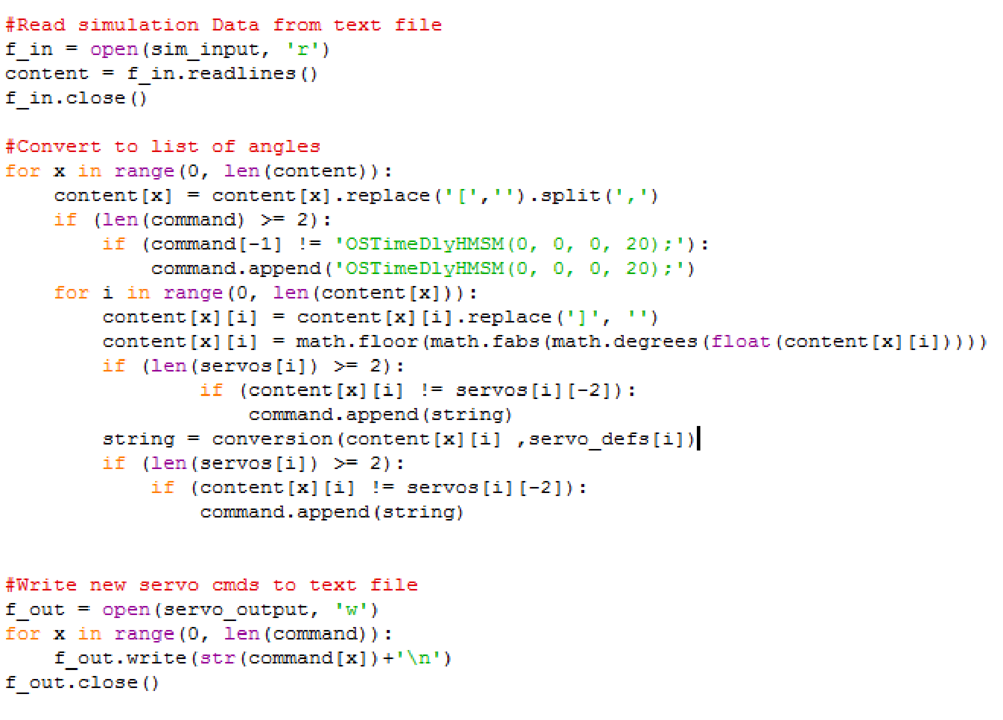
Pulse Width Modulation

- Square wave form
- Defines the angle of the servo motor



Converting from Simulation to Real World

- Slippage between gears
- Varying pulse width ranges
- Rotation around differing points
- Degree conversion for ease of use



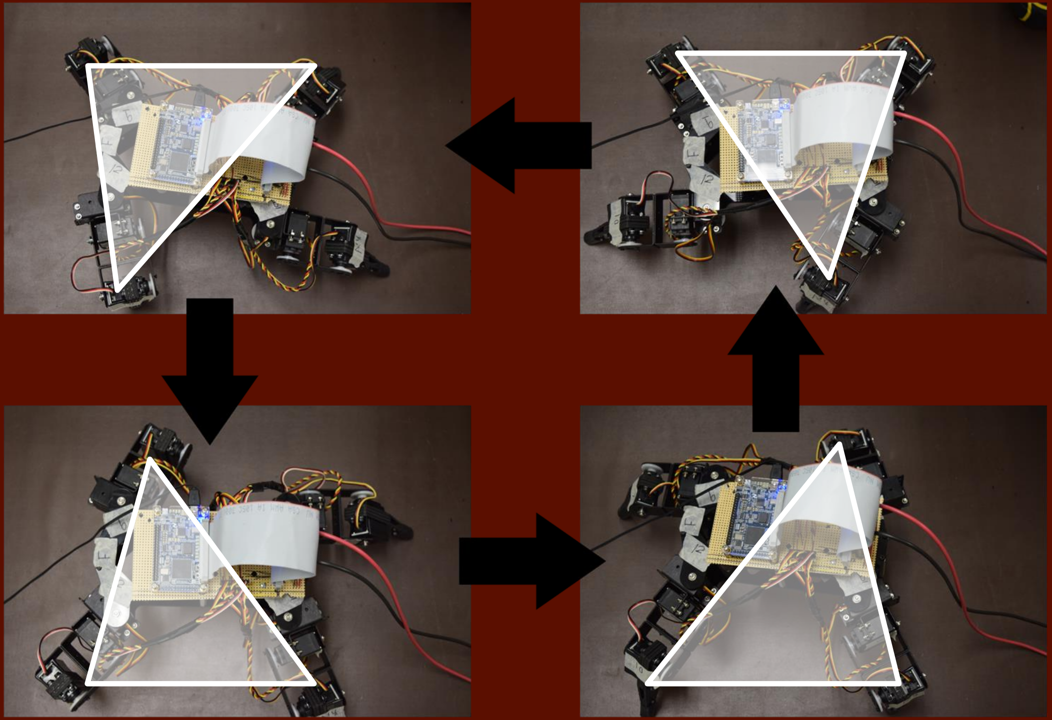
```
#Read simulation Data from text file
f_in = open(sim_input, 'r')
content = f_in.readlines()
f_in.close()

#Convert to list of angles
for x in range(0, len(content)):
    content[x] = content[x].replace('[','').split(',')
    if (len(command) >= 2):
        if (command[-1] != 'OSTimeDlyHMSM(0, 0, 0, 20);'):
            command.append('OSTimeDlyHMSM(0, 0, 0, 20);')
    for i in range(0, len(content[x])):
        content[x][i] = content[x][i].replace(']', '')
        content[x][i] = math.floor(math.fabs(math.degrees(float(content[x][i]))))
        if (len(servos[i]) >= 2):
                if (content[x][i] != servos[i][-2]):
                    command.append(string)
        string = conversion(content[x][i] ,servo_defs[i])
        if (len(servos[i]) >= 2):
            if (content[x][i] != servos[i][-2]):
                command.append(string)


#Write new servo cmds to text file
f_out = open(servo_output, 'w')
for x in range(0, len(command)):
    f_out.write(str(command[x])+'\n')
f_out.close()
```

Robot motion versus simulation motion

MakeAGIF.com

Learning to Walk Again…..

- The stances shown allow for a total weight shift from front to back legs and back again
- 3 point balance allows for a large safety margin for the torque of our servos



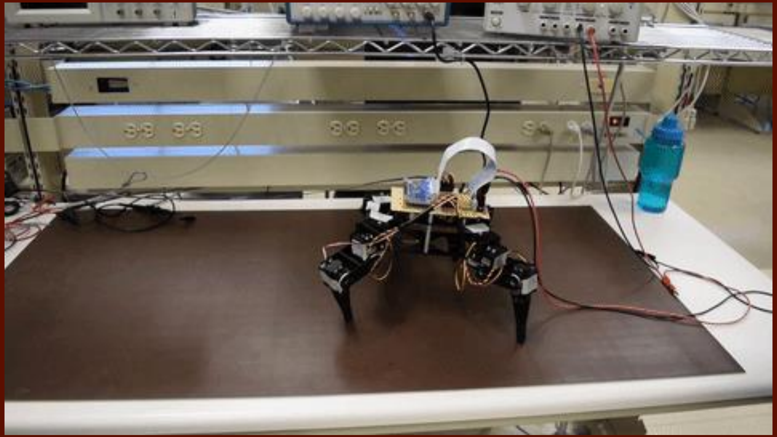
Learning to Walk Again…..

- Simulated and exaggerated weight shifting was necessary to allow for balance during steps

DEMO

Challenges

- Physical calibration
- Math and precision
- Simulation reward system to achieve a proper forward motion



Questions?