Palestine Polytechnic University

Mobile Crawler: A Robot that learns by itself

By

Buthaina Amr

Ruba Sider

Supervisor

Dr. Hashem Tamimi

College of Information Technology and Computer Systems Engineering

A project submitted in partial fulfillment for the Degree
of Bachelor's in Computer Systems Engineering

May 2018

Abstract

Reinforcement Learning (RL) is one machine learning algorithm that enables robots autonomously develop control laws and policies to achieve optimization goals by means of punishments and rewards.

The aim of this work is to demonstrate the ability of a mobile robot to learn to perform a set of actions that allow it to move forward by itself and without explicit commands. For this purpose, we constructed a crawling robot that contains 2 degrees of freedom. The robot is controlled by Arduino-Uno microcontroller. The learning process is done by Q-learning, which is a type of RL, to make the robot perform a set of trial and error actions and learns to move forward. The robot is also equipped with an ultrasonic sensor that measures the distance between the robot and the wall and provides a reward value for the robot when as it moves forward.

By allowing the robot to move its crawling arm randomly within a well-defined domain of actions and providing feedback to the Q-learning from the ultrasonic sensor, we noticed that the robot can learn the sequence of corrective actions. The experiments show that the learning process took from 5 to 10 minutes due to the random values in the Q-learning algorithm.

We propose to use this project as a learning application in the artificial intelligence course because the students can observe the process of Q-learning in an interesting way.

Acknowledgments

We express our deep sense of gratitude to our respected and learned guide to out supervisor Dr. Hashem Tamimi for his valuable help and guidance. We would like to express our deep apperception towards our Eng. Wael Takrouri for his assistance in the lab.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

Robotics is an exciting, dynamic interdisciplinary field of study.The primary task in robotics is to create an intelligent machine which performs the desired task through interaction with it is environment. The robot's behavior is acting and behaving based on knowledge and sensed information. Flexible behavior requires the ability to learn andobtain knowledge, or improve skills,adapt and reproduce tasks based on experience, observation, or instruction and autonomous learning [1].

Learning and Robotics is an attractive combination. In robotics, we wish to simulate this adaptability in the hope of being able to achieve a wide variety of tasks and rapidly learn new abilities. The purpose of our project is to work towards a reinforcement learning approach that is appropriate for robotics.

Reinforcement Learning (RL) is the ability to learn in the presence of Punishment and reward. By allowing the robot to explore the environment and perceived rewards, learning agents discover policies and controllers, often through a trial-and-error methodology [2].

we present a mobile Robot that can learn forward walking policy on basis of RL, to move forward, the robot has to repeatedly perform a cycle of moves, where a forward move gives a reward whereas any backward moves yield a punishment. The joints of the robot are driven by servos and the robot's movement is measured by the ultrasonic.

1.2 Motivation

As Robots the way of programming robots is a classical supervised way, as consequence engineers often program just a “working” behavior of a robot, but which can be far away from an “optimal” behavior. One possible solution to this general issue is offered by learning behaviors from scratch—in the same way as humans or animals do—instead of manually programming the robot. Learning in such way called trial and error. Nowadays the research domain of (RL) aims to mimic trial and error learning in machine-learning approach based on reward with the goal of maximizing the cumulative one. since robot learning is just one possible application of RL, teaching RL using crawler enabling students to observe and assess the robot’s learning behavior on different surfaces, for instance by comparing the progress of the robot through examining the learned and reward functions, or measuring the traveled distance in a fixed time interval[3].

1.3 Problem Statement

The primary goal of this project is to demonstrate the capability of a mobile robot to use it is a two-degree-of-freedom arm to perform a set of actions that allow it to drag itself forward, our crawler is controlled by Arduino Uno microcontroller. the process of learning is done using Q-Learning makes the robot learn using trial and error. the robot will serve as a platform onto which additional sensory components added such as ultrasonic, that measures the distance that indicates it is progress in moving.

1.4 Requirements

1. We need to design a two degree of freedom small size crawler that is used for educational purpose.
2. To make the crawler robot gain the ability to move forward without explicitly programming it to do so.
3. The robot should do this using a microcontroller and a distance sensor.
4. The software should have the learning concepts such as Q-Learning.

5. The robot should perform the crawling after the learning is done on a flat surface and for a range of 1-3 meters.
6. The robot should not stop or get delayed while learning or moving.
7. There should not be any human interference or external influence on the robot while learning

1.5 Literature Review

There is a vast existing literature regarding RL and robotics. This section attempts to highlight the major contributions relevant to making a Crawling Robot.

1.5.1 The Crawler, a Class Room Demonstrator for Reinforcement Learning.

They present a little crawling robot with a two Degree of Freedom DOF arm as shown in Figure 1.1 that learns to move forward within about 15 seconds in real time , the reward here is the speed of the robot, in our project we didn’t mention the time constraint, we will make the robot to take it’s time to move forward not in a specific period[4] .

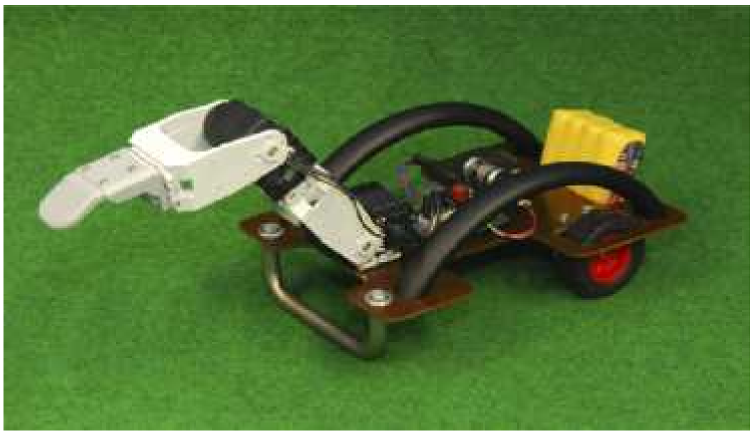


Figure 1.1: The new crawler robot for use in our laboratory tutorial,Source:[5]

1.5.2 Quadruped Robot Obstacle Negotiation via ReinforcementLearning

Describe a successful application of reinforcement learning to the problem of negotiating a wide variety of obstacles which were not seen at training time with a quadruped robot shown in Figure 1.1, including ones that had not been previously seen during training.the usedalgorithm based on a two-level hierarchical decomposition of the task, in which the high-level controller selects the sequence of foot placement positions, and the low-level controller generates the continuous motions to move each foot to the specified positions[5].

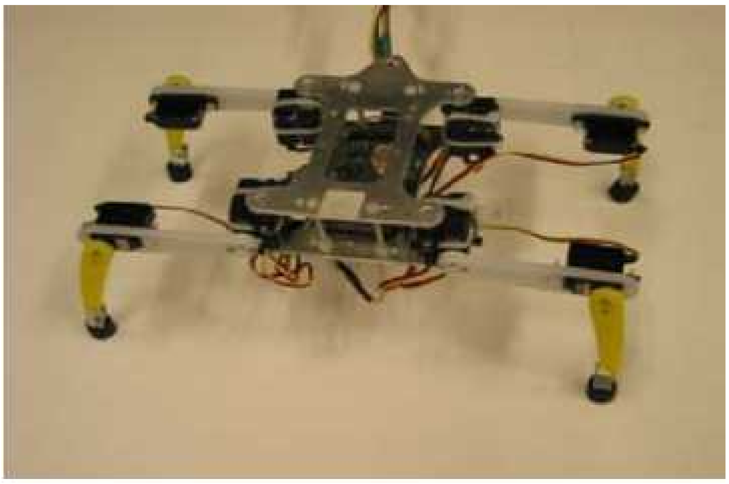


Figure 1.2: Quadruped Robot, Source: [6]

1.5.3 Robot Motor Skill Coordination with EM-based Reinforcement Learning

In Fig. 1.3 the experimental setup for the Pancake-Flipping task. A torquecontrolled 7-DOF Barrett WAM robot learns to flip pancakes in the air and catch them with a real frying pan attached to its end-effector. Artificial pancakes with passive reflective markers are used to evaluate the performance of the learned policy[7].

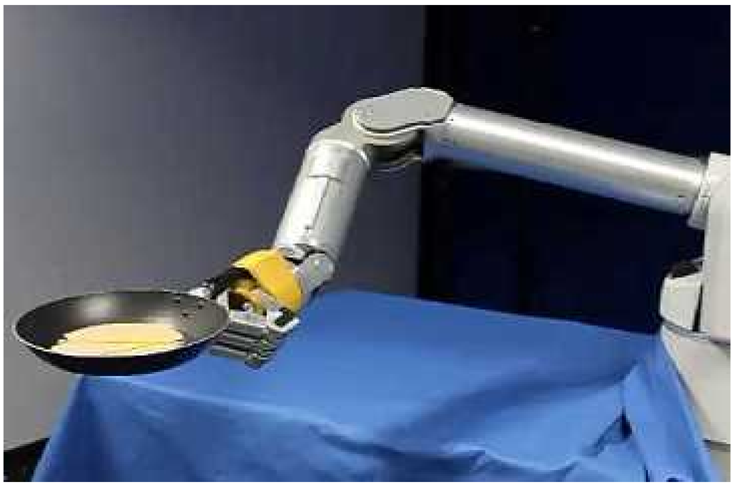


Figure 1.3Pancake-Flipping Robot, Source:[8].

Chapter 2

Background

This chapter provides a background on some of the tools used in this project. Making an Arduino Robot that teaches itself how to crawl using Reinforcement Learning algorithm called Q-Learning,it will explain the aspects of Q learning and how they are implemented in our Robot.

2.1 Q-Learning

Q-Learning is a Reinforcement Learning Algorithm that tries to find the optimal actions, where an agent in our case the crawling robot interacts with Markovian decision process (MDP). $M = S, A, R, T, \gamma$ where S is the states, A is the actions, R is the reward and γ is the learning factor, the bounded reward function is defined as $R\ (s, a) \rightarrow [R\text{min}, R\text{max}]$ where Rmin and Rmax are real numbers. The transition function T encodes the probability of reaching some next state given the current state and action. The implementation of this algorithm is specified as: [2]

Q (state, action) = R (state, action) + γ* Max [Q (next state, all actions)] …… (1)

Now we will add a similar matrix, "Q", to the brain of our agent, representing the memory of what the agent has learned through experience. The agent starts knowing starts out knowing nothing, the matrix Q is initialized to zero. The robot will explore from state to state until reaching the goal, which is moving forward, where each exploration is a new episode. The Process of learning through the experience without a teacher called unsupervised learning[9].

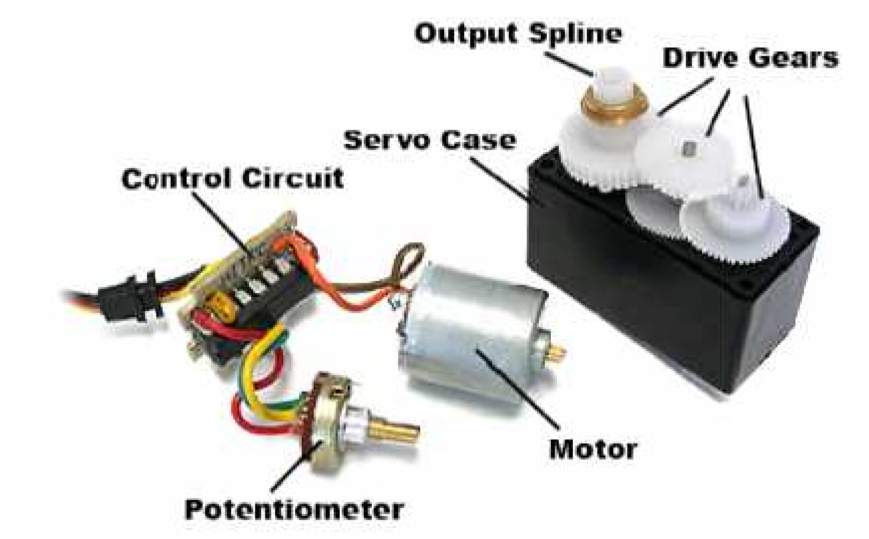
Algorithm 1 the Q learning algorithm goes as follows:

```
Start
    1. initialize the learning parameter $\gamma$
    2. Initialize the Q matrix [m]×[n]
    3. Loop for e episodes
            rand(A(states)) choose a state A randomly
            S’ = S (state, A)
        Compute Reward
            Get Qmax for the next state
            Compute Q (S, A) from the equation (1)
            S=S’
End Loop
End
```

Initializing the dimensions of the Q matrix [m]×[n] where [m] is the number of sates and [n] is the number of actions for each e episode the Q-Learning algorithm requires the rewards*r*(*s*, *a*) for each action *a* in state*s*, we simply use them to update Q matrix using the equation 1 in the previous page.

2.2 Hardware Components of the project

2.2.1 Arduino UNO



The Arduino Uno is a microcontroller board based on theATmega328P. It has 14 digital input/output pins (of which six can be used as PWM outputs), six analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an In-Circuit Serial Programming(ICSP) header and a reset button. It contains everything needed to support the microcontroller;it has 32kBytes of in system self-programmable flash program memory [10].

2.2.2Servo Motor

Servomotorprovides control of rotary position they are used extensively in the remote-control world for Aircraft, Cars,and Boats. Inside a servo case, the key components are Controlcircuitry, a motor driving reduction gears and a potentiometer driven by the reduction gears that extends through the housing and drives the actuator.An external control signal indicates the angle that the servo should move to and the motor responds by running until the potentiometer reaches the desired position that’s why it’s the most appropriate one and it comes with three wires:ground,power, and the control line.Arduino has functions available for controlling servos,Servo functions are drawn from a Servo Library Servos are positioned by setting a pin as a servo output and then sending a value from 0-180 specifying the rotation required to our project, we did not use the DC motor because it does not locate the required position[10].

Figure 2.1: Servo Motor, source [11]

2.2.3 Ultrasonic Sensor

The Ultrasonicis a sensor that measures the distance to an object by Velocity of wave propagation. it is expressed by multiplication of frequency and wavelength. The velocity of sound wave propagation in air is about 344m/ s. It measures distance by sending out a sound wave at a specific frequency and listening for that sound wave to bounce back. By recording the elapsed time between the sound wave being generated and the sound wave bouncing back, it is possible to calculate the distance between the sonar sensor and the object[12].

Chapter 3

Design

This chapter discusses the design options forboth hardware and software, the conceptual design of the system such as block diagram of the hardware component of the system.

3.1 Description of the System

This primary goal of our project is to make a crawler Robot learns to move Forward by itself without previous knowledge, this done by Setting the number of states for each servo motors then it will update the Q matrix by moving its arm. Our crawler has one Arm with two junctions, the first one moves in the x-axis and the other one moves in they-axis.

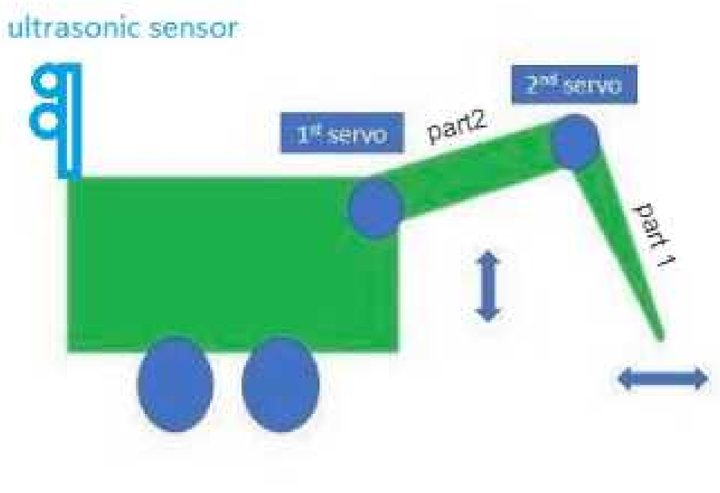


Figure 3.1: Crawling Robot Illustration.

3.2 Design Option

3.2.1 Hardware Design

As we mentioned the system consists of the ultrasonicsensor and twoservo motorsas well as the microcontroller. We will discuss each one next.

3.2.1.1 Microcontroller

We have two choices to use for our project the first choice is the Arduino UNO (AU) and the second one is Arduino mega 2560. However, Arduino Uno is a good candidate to use in our system and that because it has enough digital and analog pins than the Arduino Mega, we do not need much more memory size than the AU has, and it is cheaper than the Arduino Mega [9].

3.2.1.2 UltrasonicSensor

We have two options for the sensor we choose the first one is ultrasonic sensor and the second one is the IR sensor and we choose the ultrasonic becauseit provides a range between (1 -250 cm), In contrast, the IR has a range from (5 – 80 cm) which is not enough walking distance for our crawler to have the optimal series of actions that makes it move forward [12].

3.2.1.3 Servo Motors

For the motor that we use we have two choices as well we have Servo Motor and Stepper Motor we have chosen the Servo Motor because it is lighter, gives a greater angle precision and it is best suited to high speed than the stepper Motor [10].

3.2.2 Detailed Design

3.2.2.1 System Diagrams

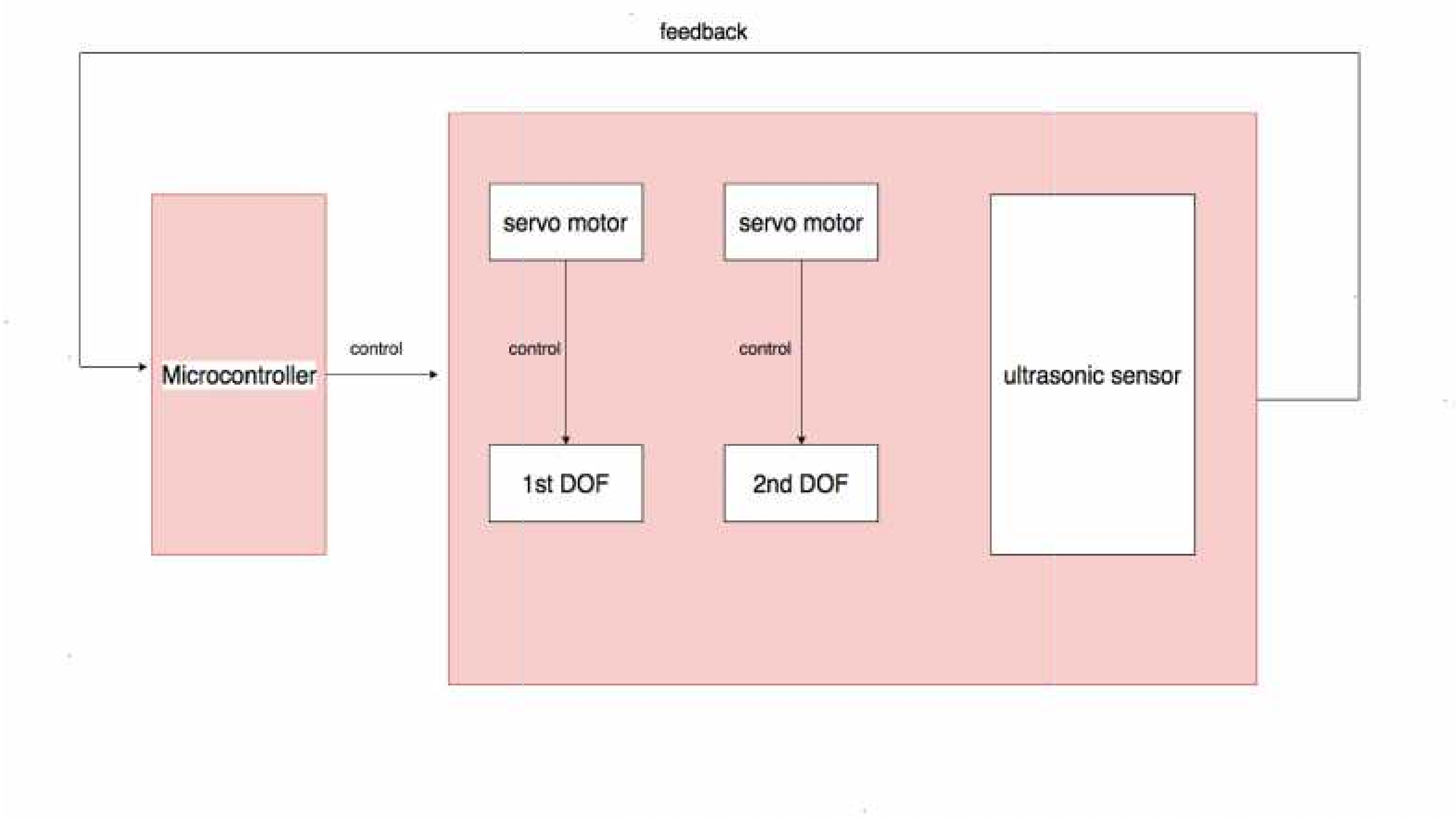


Figure 3.2: System Block Diagram

Figure 3.2: shows how the hardware sends control signals to each Servo, where each servo moves the junctions of the arm in both directional First Degree of Freedom 1st DOF and second Degree of Freedom 2nd DOF, the Agent will take a new position and the ultrasonic sensor should calculate the new distance then send a feedback to the microcontroller to decide if it's a reward or a punishment based on RL code .

3.2.2.2 System Sub-Circuits

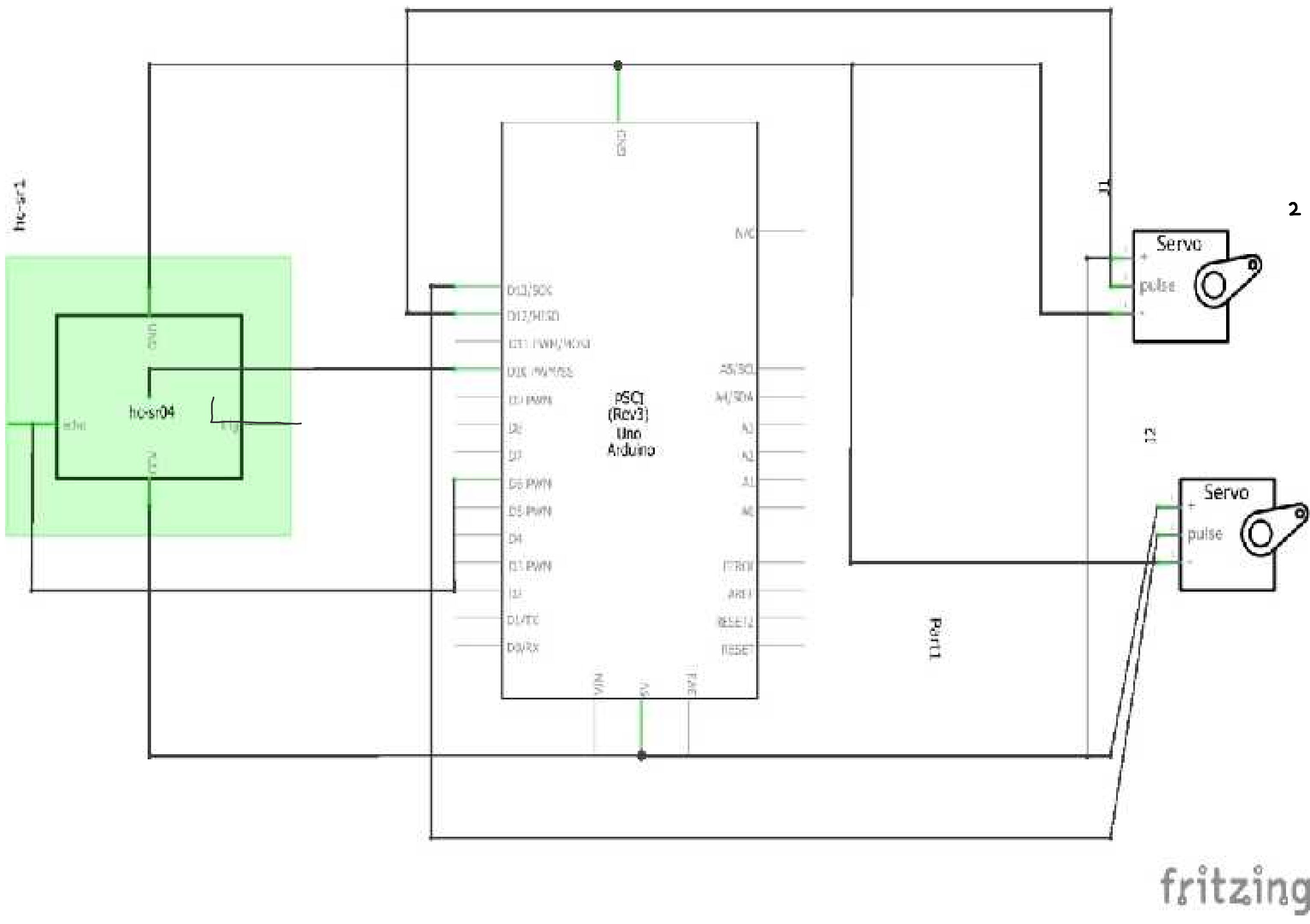


Figure 3.3: System Sub-Circuits.

Figure 3.3: System subcircuits: shows the connection of servo motors and the connection of Ultrasonic with Arduino Microcontroller.

The two servos and ultrasonic sensor are powered by Arduino microcontroller so no need for external power.Ultrasonic pins trig and echo both sending and receiving an analog wave. Servo positions are controlled by changing the values of the frequency and encoded in terms of a digital pulse.

3.2.3Mechanical Design

A three-dimensional printed arm is installed on a twowheels mobile kit, the arm consists of two parts, so we had to measure the suitable dimensions

A: The first part B: The second part

for each part of the crawler arm.Where the suitable dimensions listed as first part length is 85 mm as shown in figure 3.4.A,the second part length is 90 mm in Figure 3.4.B and figure 3.5

Figure 3.4:The crawler Parts shows the integrated parts.

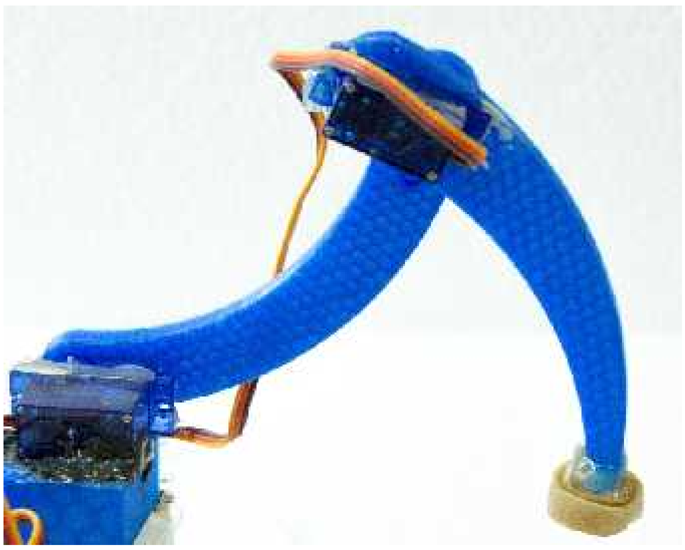
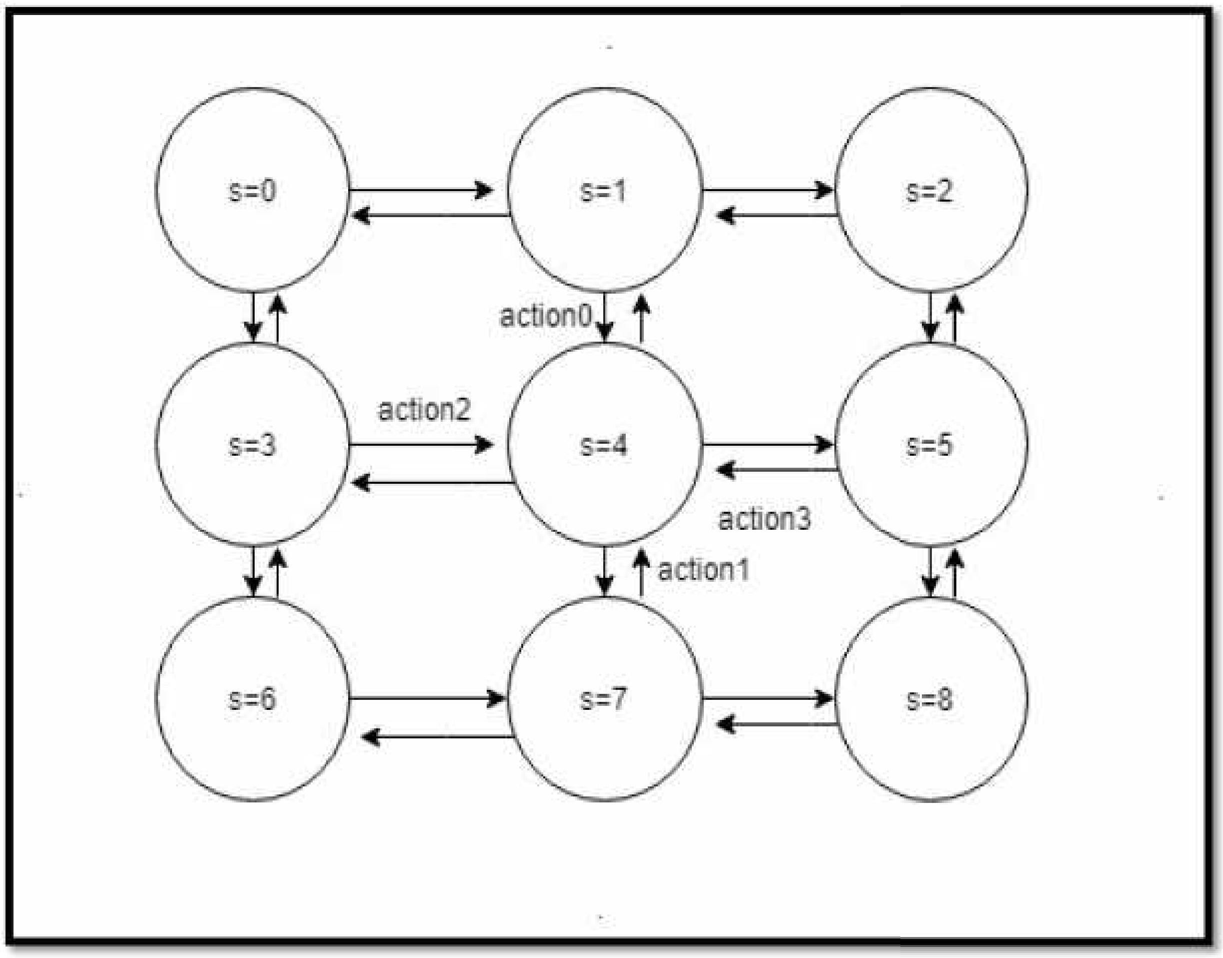


Figure 3.5: Crawler’s arm

3.2.4 Software Design

Due to the concepts of RL, the crawler should learn in terms of rewards and punishments

To get rewards the should move through states by choosing actions randomly as shown in Figure 3.6the number of arrows that come out from each state represents the number of allowed actions, so the maximum number of actions



is four.

Figure 3.6: Crawler’s action map

In Figure 3.6 the circles represent the states of the crawler for each state we have different positions for each servo motor and it moves from one state to another using action which is represented by arrows.

Algorithm 1 the Q learning algorithm goes as follows in our crawler:

Start

1. initialize the learning parameter γ

```
    2. Initialize the Q matrix [9] × [4]
    3. Loop    for e episodes
               rand(A(states)) choose a state A randomly
               S' = S (state, A)
               Implement the action
               Read ultrasonic sensor value
               Compute Reward
               Get Qmax for the next state
               Compute Q (S, A) from the equation (1)
               S=S'
End Loop
End
```

Chapter 4
Hardware and Software Implementation

This chapter demonstrates hardware and software implementation of components that interact with each other to present our crawler.

4.1 Hardware implementation

1- Arduino Uno microcontroller with servo motors

We connect the Arduino Uno microcontroller with servo motors to control the movements of them in their range. Servo motors are powered with 5 volts so no need for external power supply.we connect the control line of the first servo with pin 13 and the second servo with pin 12 as shown previously in Figure 3.3.

2- Arduino Uno microcontroller with an ultrasonic sensor

We connect the microcontroller with the ultrasonic sensor, to get the distance which represents the reward of the crawler, where we connect trig pin with pin 10 and the echo with pin 6 as shown in Figure3.3.

3- Crawler Robot design implementation

We installed a three-dimensional printed arm on the front of a two-wheel mobile Robot Kit, the arm consists of two junctions, the first part is connected to the first servo and the second part is connected to the other one.

We installed the Arduino microcontroller in the center of the kit with all concocting wires. In the back, an ultrasonic sensor was installed.

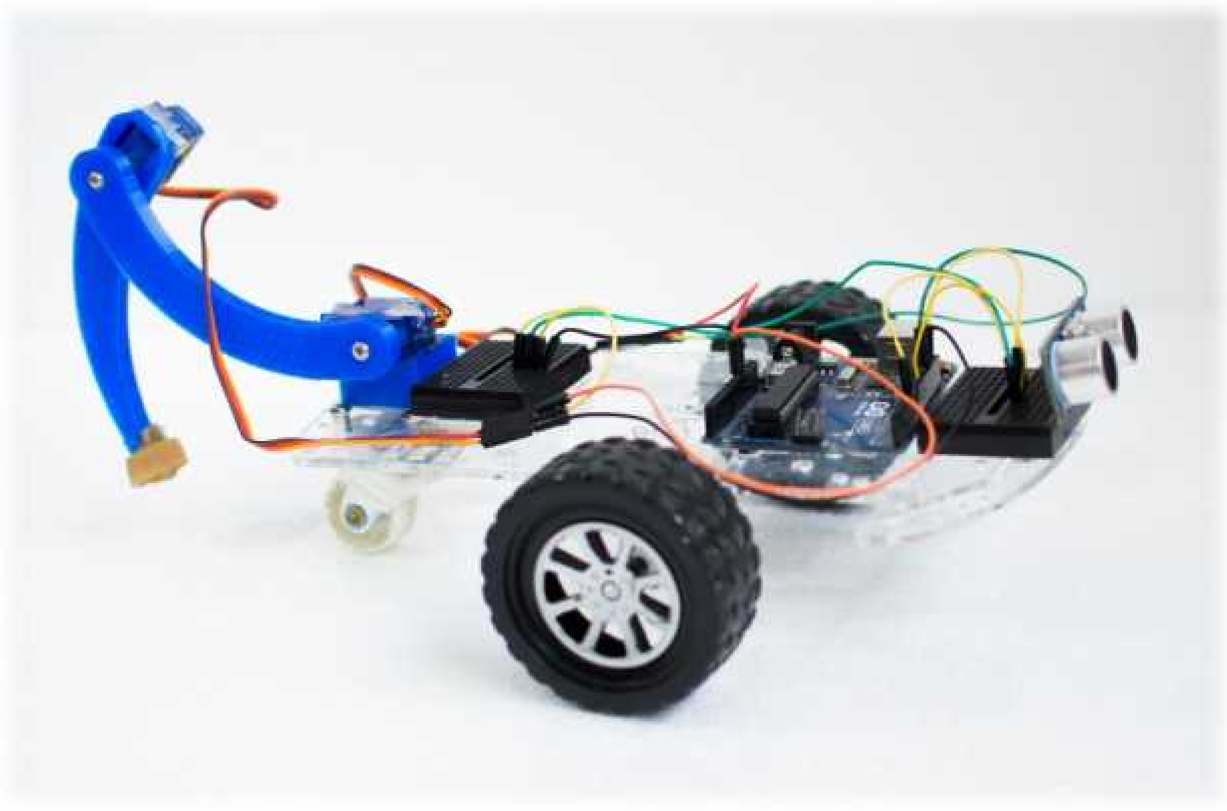


Figure 4.1: Crawler Mobile Robot.

4.2 software implementation

4.2.1 Arduino code implementation

Arduino integrated development environment (IDE) is open source software, which is cross-platform, written in the Java programming language. It runs on Windows, Mac OS X, and Linux. It is designed to introduce programming to artists and beginners who are unfamiliar with software development. An Arduino C/C++ sketch consists of two functions that are compiled and linked with a program stub main () into an executable cyclic executive program (setup () and loop ()) functions.

4.2.2 The reinforcement learning parameters

- **The Learning Factor (γ)**

In the Reinforcement Learning update equation 1 in chapter 1, Γ has a range value from 0 to 1. We have made some experiments and found that the best value for γ is 0.92. We found that if γ is close to 0 the agent will tend to consider only immediate rewards. and if it is close to one, the agent will consider future rewards with greater weight willing to delay the reward.

- The next parameter is A, which represent how much we update our utility values every time we take an action. From various recommendations of previous works, we set it to 0.1

- **The number of states**

  We set the number of states for each servo motor is 3. This means we will not allow any of the robot arms to move freely in a continuous way, but we limit it to move only within a set of three angles. The angle of the first arm can move (120,150, and 180) degrees and the second arm (8,30,60) degrees.

Since we have three states we have 3x3 state matrix and every state represents a different position of the crawler arm.

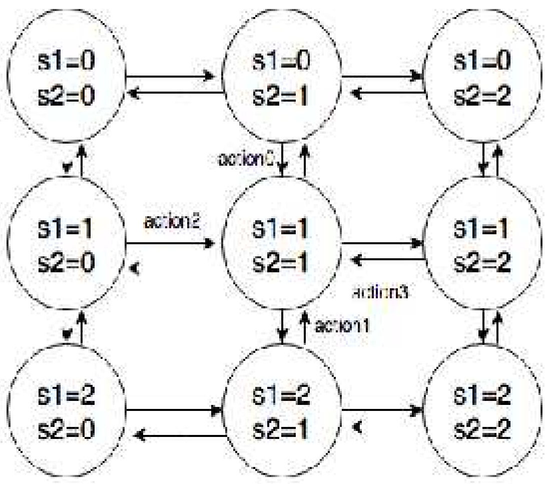


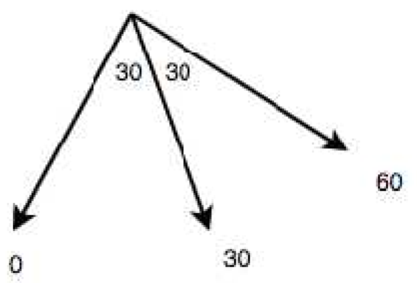
Figure 4.2: crawler state map

In Figure 4.2:shows that every state represents a different position of the crawler arm.

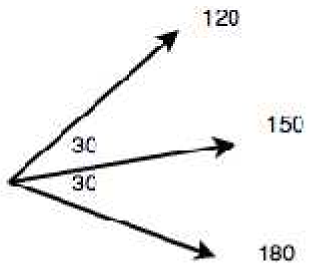
- **Implementing the physical action**

Sincewe set the number of states to 3 and the minimum and maximum angle for each part of our crawler, so the crawler should be able to recognize the change in each servo angles when an action is performed on it using this equation.

Deltaangle=(maxangle-minangle)/ (number of states-1)



A



B

Figure 4.3

Figure 4.3.: illustrates that each part of the crawler arm has a range of angles 4.3.A: the angle range for the first part of the arm that moves through (0 – 60) degree in 4.3.B: the angle range or the second part of the arm that moves from (120-180) degree.

Action	Servo	Change
Action 0	Servo 1	+30 degrees
Action 1	Servo 1	-30 degrees
Action 2	Servo 2	+30 degrees
Action 3	Servo 2	-30 degrees

Table 4:1: The Effects of Actions on the physical arm

•Reward

We implemented a simple code for the ultrasonic sensor that measures the distance first, then it takes the difference between the old distance and the new one, by doing this getting delta distance the reward will be calculated, which provides us a notation about the behavior of the Robot.in order to get better results filtering data must be included.

```
Getreward () {
NewDistance=float (ultrasonic. ping ());
deltaDistance=NewDistance-OldDistance;
if(abs(deltaDistance) <57 || abs(deltaDistance) >230) {
deltaDistance=0;}
OldDistance=NewDistance;
Return deltaDistance;
```

- **Q Learning implementations in the crawler mobile Robot Code**

First, the crawler should choose the action randomly and exclude actions that take it outside the allowed state space since the current state and the chosen action is given the crawler should be able to find the next state.

Related to the chosen action the crawler arm moves to the new position by either changing the first or the second part of the arm. A delay is needed after implementing the action, so we can give the agent enough time to move before measuring the new position to reduce errors.

We get the reward from the ultrasonic distance function that is defined previously. Then update the Q matrix based on the equation 1.

And then set s=Sprime

The crawler repeats the learning process until it finds the optimal sequence of actions.

```
Int a = getAction ();
Set Sprime
implementState(a);
```

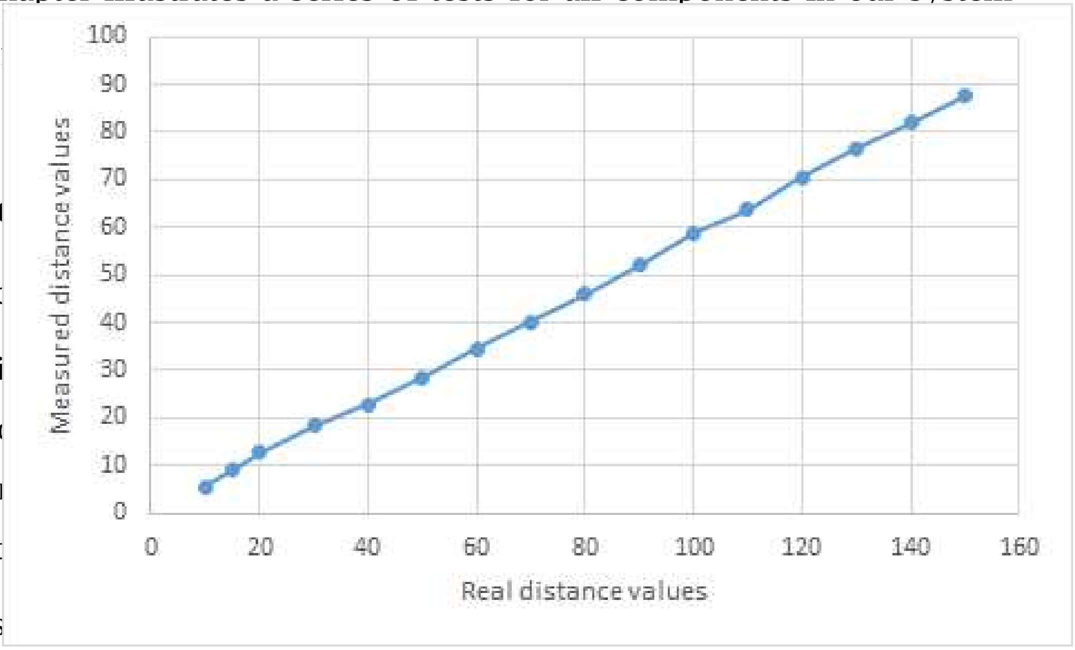
```
delay (200);

Reward =getreward ();

Q[s][a] =Q[s][a] + α * [ (R+ γ *Qmax[s’] [a’]) – Q[s][a]];

S=Sprime;
```

Chapter 5

Testing

This chapter illustrates a series of tests for all components in our system

to veri



5.1 Te

Connec

Objecti

work t

positio

neglect

Results

has angles between (0-60), servo 2 has angles (120-180).

5.2 Test two

ConnectArduino Uno with an ultrasonic sensor.

Objectives**:** connect the ultrasonic sensor to measure the distance, then we compared the measured distance to the real ones.

Results: the experiments show that Every 10 cm is considered as 6 cm.

Figure 5.1: the relationship between the real and the measured distance values

To measure the distance using Arduino microcontroller we used new ping library

That measure the distance using the following Arduino code.

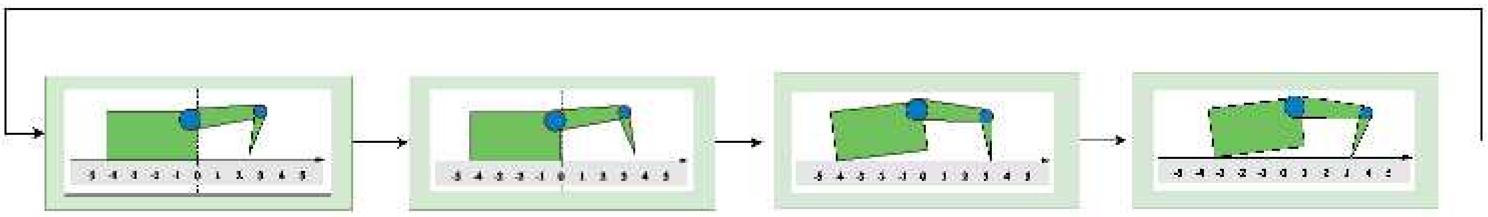
```
NewDistance = float(ultrasonic. ping());

Serial. Println (NewDistance *0.01);
```

5.3TestThree

Test the ability of the crawler Robot to crawl.

Objectives: testing if our crawler robot can move forward by giving it the optimal series of action.

Results: our Robot learns to crawl after giving it the right series of actions after getting the Range of the two servo motors.



```
VOID LOOP () {

SERVO2.WRITE(120);

DELAY (600);

SERVO1.WRITE (60);

DELAY (600);

SERVO2.WRITE(180);

DELAY (600);

SERVO1.WRITE (0);

DELAY (600);
```

(A) (B) (C) (D)

Figure 5.2: Example of the optimal set of actions

Where:

(A) Move the second part of the crawler arm by setting the second servo to 120 degrees

(B) Move the first part of the crawler arm by setting the first servo to 60 degrees

(C) Move the second part of the crawler arm by setting the second servo to 180 degrees

(D) Move the first part of the crawler arm by setting the first servo to 0 degree

5.3 Test Four

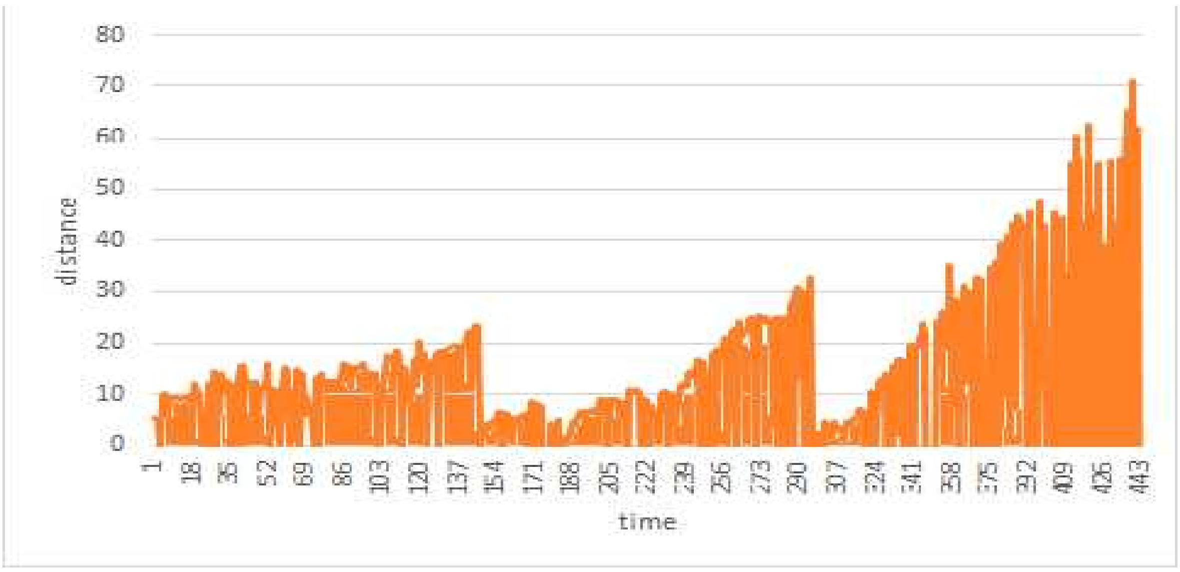
Test the results

Objectives: testing the learning speed of the crawler through the passage of time

We decompose the values into three parts each part represents one minute.

After each minute we take the difference between the current distance and the last distance value in the first part.

Results: In the first and second minute the learning speed is slow that illustrates that the robot has not learned yet, last minute the learning speed is fast and that indicates that the robot has learned as shown in Figure 5.3.



6.1 Project Outcomes

Figure 5.3: learning process through time

Chapter 6

Conclusion

6.1 Project Outcomes

This project presents a crawling robot asan application for reinforcement learning. The robot learns a forward-walking policy from scratch from (5-10) minutes of reinforced sensorimotor interactions. The state space consists of 9 states and maximum 4 actions with Q matrix size 4*9 and Γfactoris 0.92.

6.2Challenging Faced

- Ultrasonic Range Sensor Accuracy

Ultrasonic ranging module HC - SR04 provides 2cm - 400cm non-contactmeasurement function, the ranging accuracy can reach to 3mm, since this we had accuracy issues for measuring the distance. So much noise appeared.

Taking delta distance instead of the direct distance,this reduces the sensor noise.

- The fraction of the crawler arm with the ground

  As a first task, a flat surface is given. On such a ground, the robot moves in an effective way,an elastic joint is placed on the arm. Whenever the robot lifts its body a little bit, it can move effectively.

6.3 Future Work

Since this is an ongoing project we currently implementQ-Learning on the hardware robot. Once it is implemented we suggest making the crawler Robot as a didacticinstrument in laboratoriesthat will enable students to observe and assess the robot learning behavior on a differentsurface.

They can perform their own experiments on the robot application, for instance by comparing the progress of the robot through examining the reward functions or measuring travels distance in a time interval.

References

[1] Gaskett, C. (2002). Q-Learning for Robot Control [Pdf]. p.1.

[2]: Cutler, M. J. (2015). Reinforcement Learning for Robots through Efficient Simulator Sampling [Pdf].

[3] Tokic, M., Usadel, A., Fessler, J., & Ertel, W. (2012). On an educational approach to behavior learning for robots. Scholar Articles on an Educational Approach to

Behavior Learning for Robots. Retrieved December 2017, from http://www.tokic.com/www/tokicm/publikationen/papers/rie2010.pdf

[4] Tokic, M., Ertel, W., & Fessler, J. (2009). The Crawler, A Class Room Demonstrator for Reinforcement Learning. In FLAIRS Conference (pp. 2471-2482).

[5] Tokic, Ertel, & Fessler. (n.d.). Figure 1.1: The new crawler robot for use in our laboratory tutorial [The new crawler robot for use in our laboratory tutorial]. Retrieved from https://www.researchgate.net/publication/221438658_The_Crawler_A_Class_Room_Demonstrator_for_Reinforcement_Learning

[5] Lee, H., Shen, Y., Yu, C., Singh,& Y. Ng, A. Quadruped Robot Obstacle Negotiation via Reinforcement Learning. Retrieved March 2018, from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.446.8818&rep=rep1&type=pdf

[6] Lee, H., Shen, Y., Yu, C., Singh,& Y. Ng, A,Quadruped robot [Digital image]. (n.d.). Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.446.8818&rep=rep1&type=pdf

[7] Kormushev, P., Calinon, S. and Caldwell, D.G. (2010)
Robot Motor Skill Coordination with EM-based Reinforcement Learning
In Proc. of the IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS), Taipei, Taiwan, pp. 3232-3237.

[8] Kormushev, P., Calinon, S., & Caldwell, D. (n.d.). Pancake-Flipping Robot [Experimental setup for the Pancake-Flipping]. Retrieved from https://kormushev.com/papers/Kormushev-IROS2010.pdf

[9] Q-Learning. Retrieved December 11, 2017, from http://mnemstudio.org/path-finding-q-learning-tutorial.htm

[10] Boxall, J (2013). Arduino workshop. San Francisco

[9] Stepper vs Servo. Retrieved December 11, 2017, from https://www.amci.com/industrial-automation-resources/plc-automation-tutorials/stepper-vs-servo

[11] internal circuit, servo motor, digital image, from https://www.mysensors.org/build/servo.

[12] Http://www.symmetron.ru/suppliers/murata/files/pdf/murata/ultrasonic-sensors.pdf [PDF]