An Outreach Engagement Activity

Student Number: C1221401
Degree Scheme: Computer Science with Security and Forensics
Year: 3 2015/16
Supervisor: Helen Phillips
Student Name: Shanay Shah

I. ABSTRACT

This dissertation aims to create an outreach engagement activity for the Cardiff School of Computer Science in which the main goal is to have the students interact with hardware such as the Raspberry Pi and Arduino rather than just the traditional outreach activities that only involve software. Through this dissertation, an engagement activity will be created to run for one to two hours and also be suited for groups of students up to thirty. Through this project, I will make a vehicle-based robot using the Arduino and create an outreach activity based on the development of this robot. Furthermore, the activity will go on to use the Raspberry Pi with the Sense HAT add on.

II. ACKNOWLEDGEMENTS

I am also grateful to Helen Phillips, my supervisor, in the Department of Computer Science and Informatics. I am extremely thankful and indebted to her for sharing expertise, and sincere and valuable guidance and encouragement extended to me.

I take this opportunity to express gratitude to all of the Department faculty members for their help and support through out the university degree which allowed me to get this far. I also thank my parents for the unceasing encouragement, support and attention.

III. TABLE OF CONTENTS

I. ABSTRACT II

II. ACKNOWLEDGEMENTS III

III. TABLE OF CONTENTS IV

IV. LIST OF FIGURES VI

1. INTRODUCTION 1

1.1. Goals and Motivation 1

1.2. Approach 1

1.3. Outcomes 3

2. BACKGROUND 4

3. RESEARCH REVIEW 5

3.1. Introduction 5

3.2. Products On the Market or Coming to the Market 5

3.2.1. Codie — Cost: $240 — Target Age Group: 8 - 12 5

3.2.1. Play-I (Dash) — Cost: $149 — Target Age Group: 8+ (Choi, J. 2014) 5

3.2.2. OZOBOT — Cost: $59.95 — Target Age Group: 8+ (Choi, J. 2014) 6

3.2.3. Cubelets — Cost: $159.95 — Target Age Group: 4+ (Booth, H. 2014) 6

3.2.4. Lego Mindstorm — Cost: $349 — Target Age Group: 10+ (Choi, J 2014) 6

3.2.5. Conclusion 7

3.3. Raspberry Pi and the Arduino 7

3.3.1. Power Requirements 7

3.3.2. Networking 8

3.3.3. Sensors and other Peripherals 8

3.4. Arduino and Raspberry Pi System 8

3.4.1. Serial Peripheral Interface (SPI) 8

3.4.2. Inter-Integrated Circuit (I2C) 9

3.4.3. Serial Communication 10

3.4.4. Conclusion 10

3.5. Sensors 10

3.5.1. Ultrasonic Distance Sensor 10

3.6. Motors 11

3.6.1. DC Motors 11

3.6.2. Servo Motors 11

3.6.3. Stepper Motors 12

3.6.4. Conclusion 12

3.7. Chassis 12
4. SPECIFICATION, DESIGN and TESTING 13
4.1. Introduction 13
4.2. Robot 13
4.2.1.Iteration 1 13
4.2.1.1.Requirements 13
4.2.1.2.Design 13
4.2.1.3. Test - Powering Two or Four Motors 17
4.2.1.4.Test - Iteration 1 23
4.2.1.Iteration 2 23
4.2.1.1.Requirements 23
4.2.1.2.Design 23
4.2.1.3.Test - Distance Sensor 24
4.2.1.4.Test - Iteration 2 31
4.2.1.Iteration 3 32
4.2.1.1.Requirements 32
4.2.1.2.Design 32
4.2.1.3.Test - Iteration 3 33
4.3. Engagement Activity 34
5. EVALUATION 35
6. FUTURE WORK 37
7. REFLECTION 38
8. APPENDICES 41
8.1. Appendix A - External Material 41
8.2. Appendix B - Sense HAT Worksheet 42
8.3. Appendix C - Sense HAT Fact Sheet 49
8.4. Appendix D - Matching National Curriculum Criteria 52
8.5. Appendix E - Teachers Feedback Form 54
8.6. Appendix F - Students Feedback Form 55
8.7. Appendix G - Arduino Worksheet 56
9. REFERENCES 65
10. Bibliography 68

IV. LIST OF FIGURES

Figure 1: Waterfall Model — http://www.umsl.edu/~hugheyd/is6840/waterfall.html 1

Figure 2: Agile Model — http://strategybeach.com/our-agile-development-methodology/ 2

Figure 3: HC-SR04 — http://microcontrollerelectronics.com/distance-sensing/ 10

Figure 4: DC Motor — http://proveedor.com.pk/DC-Motor-Gear-Wheel 11

Figure 5: Servo Motor — http://stm32f4-discovery.com/2014/10/library-42-control-rc-servo-stm32f4/ 11

Figure 6: Stepper Motor — https://simple.wikipedia.org/wiki/Stepper_motor 12

Figure 7: H-bridge Circuit — http://www.modularcircuits.com/blog/articles/h-bridge-secrets/h-bridges-the-basics/ 14

Figure 8: H-bridge Circuit Current Flow Forwards Direction 14

Figure 9: H-bridge Circuit Current Flow Backwards Direction 15

Figure 10: H-bridge Pin Diagram — http://www.rakeshmondal.info/L293D-Motor-Driver 15

Figure 11: Schematic for Connecting DC Motors to Arduino via H-bridge 16

Figure 12: Sound Wave of the Motor Receiving 1/4 of the Power 18

Figure 13: Sound Wave of the Motor Receiving 1/2 of the Power 18

Figure 14: Schematic for Powering Motors Directly from Power Supply 19

Figure 15: Code using DCMotorBot Library 20

Figure 16: Code Used to Rotate Robot in Left Direction 20

Figure 17: Code to Test Two-Wheel Based Robot 22

Figure 18: HCSR04 to Arduino Connection 24

Figure 19: Schematic for Test of HC-SR04 25

Figure 20: Code for Test of HC-SR04 25

Figure 21: Output from Serial Monitor for Test of HC-SR04 26

Figure 22: Function for Scanning for Obstacles 26

Figure 23: Code for Obstacle Detection and Stopping Robot 27

Figure 24: Schematic for Connecting Motors and HC-SR04 to Breadboard 28

Figure 25: Functions Created for Robots Movement in all Directions 29

Figure 26: Code using Self Made Functions for Robot Movement 30

Figure 27: New and Revised Scan Function 31

Figure 28: Navigation Function 32

1. INTRODUCTION

1.1. Goals and Motivation

The goal of this project is to develop and create an application that can be used by the Cardiff School of Computer Science and Informatics outreach programme to schools, the client requested the development of something tangible that linked hardware as well as software, something other than a game on a computer screen. The application would be used to teach students currently studying the key stage three curriculum the fundamentals of computer programming, in the effort to further their interest and knowledge of the opportunities averrable to them. Beyond that the client just wanted to find out what could be achieved with the Raspberry Pi or/and the Arduino.

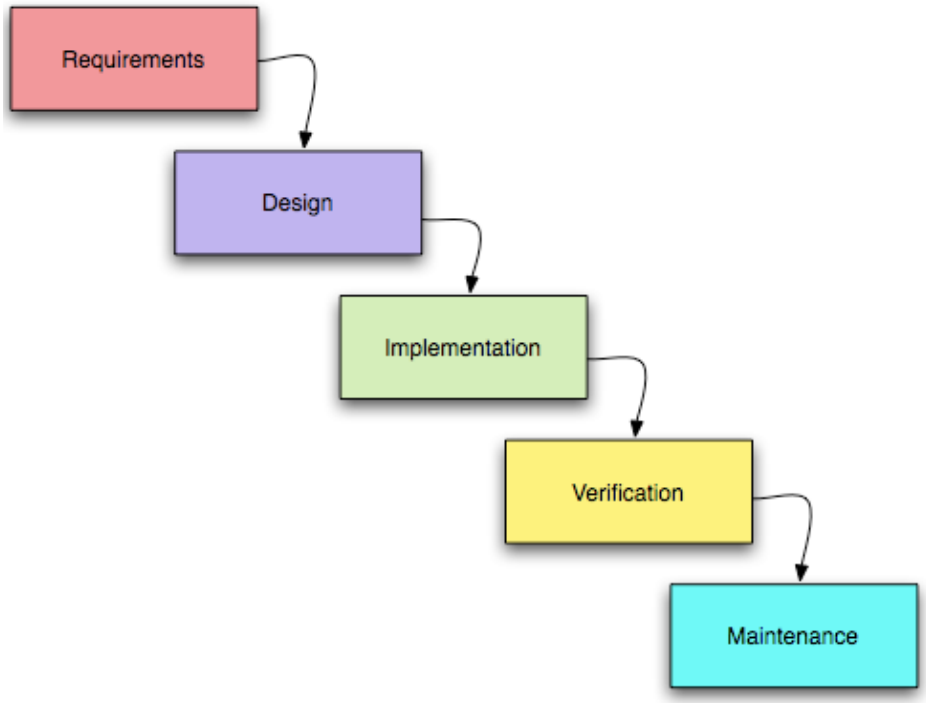
1.2. Approach

To decide on which approach to use in order to tackle this problem I chose to research into two different models that I thought would be best suited to the project. The first is the traditional waterfall method. With this approach, the entire process is divided into separate phases. Using this approach would mean that in order to start a new phase of the project, the previous phase must have been completed. The situations in which the use of the waterfall method is recommended are as follows **(Tutorials Point A. 2016)**:

- Requirements are clear, fixed and well documented.
- The definition of the product is stable.
- The technology used is static and understandable, not dynamic.
- Abundant resources with the needed expertise are available for product support.
- The length of the project is short.

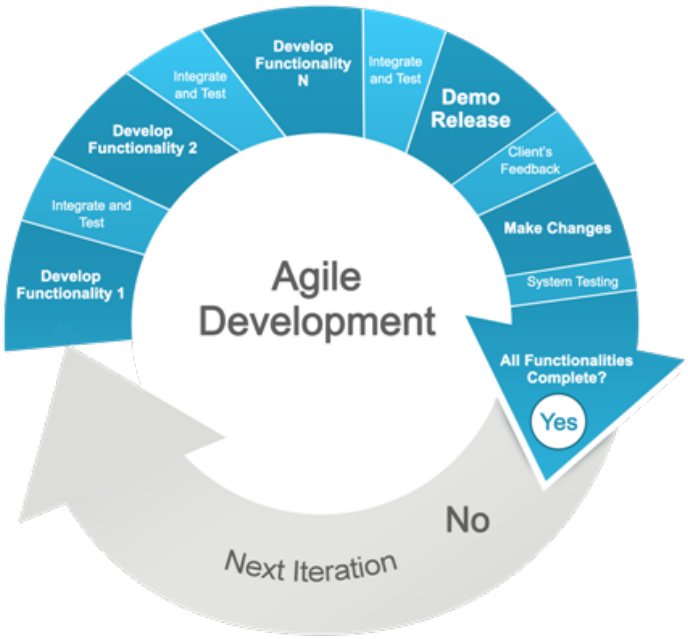
The waterfall model is extremely easy to understand and use as it allows for the project to be broken down into individual phases which can be easily controlled. The development cycle has six phases and is pictured below:

FIGURE 1: WATERFALL MODEL — http://www.umsl.edu/~hugheyd/is6840/waterfall.html



The alternative method I thought may suit the project was the agile development model which combines an iterative and incremental development process **(Tutorials Point B. 2016)**. This results in miniature incremental releases where each release builds on previous functionality. It is an extremely realistic approach compared to the waterfall method however it is not ideal for handling complex dependencies. Even though there is a lack of detailed planning and an overall plan must exist in order for it to work. A figure describing the agile approach is featured below:

FIGURE 2: AGILE MODEL — http://strategybeach.com/our-agile-development-methodology/



As you can see the waterfall model relies on a predictive approach whereas the agile model is founded on more adaptive software development models. Therefore, there is no detailed planning when using the agile development model and as product requirements change you have to adapt to them dynamically.

Initially, the waterfall model seemed appropriate and an initial plan was made with that in mind. This is because the requirements were fixed for example 'build moving vehicle/robot'. Furthermore, the product (robot) had a stable definition and finally the length of the project was short. Therefore, my initial plan was created in accordance with the waterfall model and the project had been broken down into several small phases which could easily be controlled.

However as the first couple of weeks of the project went by I came to realise that I had chosen the wrong development model and that the agile model was greater suited to the project. This is due to several reasons, firstly, there was a lack of resources and the expertise needed to gain from these resources efficiently. The lack of expertise meant that I had to research into an unknown field and then order the components and parts required to build the robot. Due to delays in the shipping of certain parts or due to the location they were being shipped from, often I could not keep up with the plan that had been structured initially which followed the waterfall model. Therefore, I had to change my approach and adapt to these dynamic conditions which made me realise that the agile model of development was, in fact, better suited to this project.

Additionally, in a meeting with my supervisor, she showed me the Astro Pi Sense HAT which is an add-on board for the Raspberry Pi which has a number of sensors and an LED matrix built in. After experimenting with the board I realised that using it would be ideal for teaching the students some programming basics whilst showing them some real world applications. In this way, the students would be able to realise the applications of what they are taught which can create a deeper understanding and interest in the topic.

1.3. Outcomes

The client (Helen Phillips) ran a teacher advisor panel in June 2014 and the list below are the things teachers asked for in an outreach activity:

- An inexpensive engagement activity, so that schools would be able to afford it
- An activity that the student can carry out themselves and create a project out of it
- An activity in which the students are able to gain skills such as team work, problem solving, programming and testing.
- An activity that uses a combination of both hardware and software.

More specifically the outcome will be a vehicle-based robot that uses the Arduino along with the relevant material in order to begin teaching the basics of programming using the robot and teaching material that can be used with the Raspberry Pi and the Sense HAT.

2. BACKGROUND

Robots are able to capture a child's imagination by creating a fun, physical and interactive learning process. Children are able to learn the basic fundamentals of programming by interactively playing with robots and getting them to move through various sequences. They are often controlled from intuitive applications that are specifically made for programming the robot using a visual programming language. Moreover, as the children interact with robots they are able to develop key components of programming knowledge and computational thinking. Even through the programming will largely be visually based it will allow for the children to understand textual programming languages with greater ease at a future date. With which they will be able to execute highly complex missions on computer systems **(Geer, D. 2014)**.

The idea of robots existing within a classroom may have seemed extremely unrealistic not too long ago, but with the pace that technology is moving at, there are an increasing number of products on the market or coming to the market that can be used to teach children computer programming. *Schools are experimenting with robots to see if they are able to add any value to the classroom.* It is important to realise that these robot teachers are not a replacement for classroom teachers, however, they are simply tools that can be used by schools and teachers to further the students interest to learn. This method of teaching is not yet a widely adopted, as these products often come at a high cost and many educational institutions are unable to budget for them. Therefore, many students are unable to experience learning how to program in a fun and interactive manner. Which is why this project will aim to build a relatively more affordable alternative to the existing products on the market.

In order to create awareness of the various Computer Science related degrees offered at Cardiff University this project intends to develop a robot that can be used as an aid in teaching the fundamentals of programming, it will be necessary to identify the key aspects of teaching that need to be addressed, the components needed in order to create the robot and methods that can be used to interface them.

3. RESEARCH REVIEW

3.1. Introduction

This section of the report will discuss which products exist on the market or are coming to the market in the new future that can be used to teach children aspects of computer programming as well as past and currently on-going developments and research regarding creating a robot using a Raspberry Pi and an Arduino. Furthermore, this section will discuss the components available and why certain components were chosen over others.

3.2. Products On the Market or Coming to the Market

3.2.1. Codie — Cost: $240 — Target Age Group: 8 - 12

http://www.getcodie.com/

Codie was created by the Hungarian startup Codie in 2013 **(Facebook, Codie. 2016)**. In essence, it is a programmable robot that is controlled via a smart phone application that teaches kids the principles of coding. It does this by using a visual programming language that has been simplified into draggable blocks therefore making it easy and fun for kids to play with. Codie is aimed at a target audience ranging from eight to twelve year olds **(Lomas, N. 2015)**.

By connecting to Codie through its companion application and a bluetooth connection kids are able to go through exercises which teach them the basic fundamentals of programming by watching the robot perform various actions and reactions. These actions and reactions are due to the seven on board sensors which include a temperature sensor, ultrasound distance sensor, light sensors, and a line tracking sensor. Furthermore, it includes a buzzer, a microphone and LED lights for further output and input **(Lomas, N. 2015)**.

3.2.1. Play-I (Dash) — Cost: $149 — Target Age Group: 8+ **(Choi, J. 2014)**

https://www.makewonder.com/

Play-I make a range of robots that introduce programming to children as young as five years old. I will be focusing on the Dash robot that they make as it is most similar to the one I have created. It has abilities to respond to voice commands, navigate, and avoid objects, dance, and sing. There are several applications you can use to program the robot, popular ones being Wonder and Blockly

(**Make Wonder. 2016)**. Play-I has created a number of accessories that can be snapped on to the robot which will give it some extra abilities. For example, there is a xylophone add-on, with which the students can learn some programming. The application will ask the children to get the robot to play certain notes, for explanation purposes lets say this is 3 red notes and 2 blue notes, the child must get the hammer to drop three times and then move to be over the blue note and drop another 2 times **(Biggs, J. 2014)**.

3.2.2. OZOBOT — Cost: $59.95 — Target Age Group: 8+ (Choi, J. 2014)

http://uk.pcmag.com/robotics-automation-products/8197/news/tiny-60-ozobot-toy-robot-plays-ipad-table-games

Ozobot is different to the other robots available on the market mainly due to its size. Where other robots are big and often controlled or programmed with smartphones, Ozobot is a tiny one inch robot that is designed to be used on top smartphone, tablet screens and on paper **(Tracy, 2015)**.

3.2.3. Cubelets — Cost: $159.95 — Target Age Group: 4+ (Booth, H. 2014)

http://www.modrobotics.com/cubelets/

Building a robot using Cubelets is like building blocks **[Peterlee. 2015]**. Each cube is an individual robot and they can connected to each other to bring additional functionality to the robot. There are three different types of blocks available offering functions including Think, Action and Sense. The Think blocks have logic and math functions that allow it to give commands to the other blocks. The Action blocks then perform the commands given to them and the Sense blocks are used to sense the environment around the robot. Chubbiest has developed an operating system that allows the cublets to communicate with each other as well as other connected devices. The Cubelets smartphone application is designed to work using the operating system and can be used to read and write to the cublets.

3.2.4. Lego Mindstorm — Cost: $349 — Target Age Group: 10+ (Choi, J 2014)

http://shop.lego.com/en-US/LEGO-MINDSTORMS-NXT-2-0-8547

Lego Mindstorm sets are flexible robotic kits that use a combination of sensors, motors and an electronic brain to give anyone the ability to make a robot. Included in the kit there is the actual

chip that functions as the 'brain' of the robot, three servo motors, a colour sensor, a touch sensor, an infrared sensor and 550 LEGO Technic pieces. Users are able to build a variety of remote controlled robotic devices with this kit. The Mindstorm uses a free software called LabView **(Burns, C. 2014)**. It is relatively easy to get to grips with and allows for control of all the moving parts and reaction to certain stimuli.

3.2.5. Conclusion

As you can see the products that are already on the market or coming to the market in the near future have some extremely advanced capabilities that allow them to really interact with the student on multiple levels of complexity. Most of them are above the $150 price mark and far more suited for one robot per user, meaning they are relatively expensive pieces of equipment. Hence schools may find it difficult to use these as a tool in teaching due to the high price marks. Furthermore they are not all targeted for the key stage three age group. However with the increasing amount of competition in this new and growing market means that soon there will be a greater variety of alternatives on the market that could be more affordable for schools. Once I had conducted this research I had a feeling that with the resources, time and expertise I had at my disposal it would be extremely difficult to make a robot that could compete with the functionality of those discussed above. However I went ahead with the plan as proposed.

3.3. Raspberry Pi and the Arduino

The Raspberry Pi and Arduino were both originally created to be teaching aids, therefore they are relatively easy to get to grips with. Which in turn leads to their ever increasing popularity across all age groups. The Raspberry Pi was invented and designed in the United Kingdom by Eben Upton and some of his colleagues at the University of Cambridge's Computer Laboratory (**Raspberry Pi, About Us)**. The Arduino on the other hand was developed and designed in Italy and gets it name from the bar where Massimo Banzi and his cofounders first came up with the idea (**Kushner, D. 2011)**.

For all intents and purposes, the Raspberry Pi is a fully functional computer. It has all the frills of a computer, with a dedicated processor, memory, and a graphics driver for output through HDMI. It even runs on a specifically designed version of the Linux operating system which makes it easy to install most software available for Linux. The Pi does not have any internal storage therefore you must use external storage in the format of SD cards as flash memory for the entire system. As the Pi has independent network connectivity it makes for easy access through secure shell (SSH) and file transfer protocol (FTP) can be used to transfer files to the device **(Bourque, B. 2015)**.

Arduino's are micro controllers and do not share the frills of a computer. They simply execute code that is passed through them as the firmware interprets it. Therefore it does not have a full operating system running on it hence you loose the basic tools that operating systems provide. But this makes the execution of the code simpler and there is no operating system overhead. The Arduino's key purpose is to interact with sensors and other peripheral devices **(Bourque, B. 2015)**.

3.3.1. Power Requirements

The two systems have completely different power requirements. The Raspberry Pi requires a constant five volts of power to operate and should shutdown via a software process like a normal

computer. Whereas the Arduino simply starts executing code once it is switched on and stops when the power supply is pulled out. Portability is an concern with the Pi as it requires more than just a couple of AA batteries to power it, you need a power supply and some additional hardware in order to provide it with the constant power it needs. The Arduino only requires a battery pack that keeps the voltage above a certain level. Even if the power input drops and the Arduino does not have sufficient power to operate, there is no risk of ending up with a corrupt operating system or any other software based errors. It will simply start running the code again once it acquires the power needed **(Bourque, B. 2015)**.

3.3.2. Networking

The Raspberry Pi has a built in ethernet port for easy network access and in order to achieve wireless connectivity on the Pi all thats required is a WiFi dongle which can be sourced easily and bought cheaply. The Arduino unfortunately was not built for plug and play network access it requires some additional tinkering, but it is possible **(Bourque, B. 2015)**.

3.3.3.Sensors and other Peripherals

When it comes to sensors both systems have a number of interface ports, but it is easier to connect analogue sensors to the Arduino as it can easily interpret and react to large range of sensor data. Whereas the Pi requires additional software to effectively interface with analogue sensors. The Pi can interface with digital sensors right of the mark without the requirements for additional software **(Bourque, B. 2015)**.

3.4. Arduino and Raspberry Pi System

While there are many I/O boards available or under development for the Raspberry Pi, the Arduino is well established. The Arduino has built in capabilities such as analog inputs and pulse width modulation (PWM) outputs which do not exist on the Raspberry Pi. Additionally the Arduino provides multiple timers, interrupts and other such features which allow for precise real time processing. However the Raspberry Pi has more processing power and hence can dish out directions and instructions whilst the Arduino can be used as the sensory workhouse. The Arduino is better suited for motor driving and sensor reading where you can have a network capable Pi controlling and driving it. There are several ways of interfacing the Raspberry Pi and the Arduino these include USB Serial, Transistor-Transistor Logic (TTL) Serial, Inter-integrated Circuit (I2C) or Serial Peripheral Interface bus (SPI).

3.4.1.Serial Peripheral Interface (SPI)

This is a synchronous serial data protocol that is used by micro controllers to allow them to communicate with one or more peripheral devices quickly over short distances. Furthermore it can be used for communication between two micro controllers. SPI is a very well established chip-chip communication method that is implemented in hardware on both the Pi and the Arduino. Devices communication using SPI are in a master-slave relationship **(Circuit Basics A. 2016)**. The master is the controlling device in our case the Raspberry Pi and the slave is takes instructions from the master in our case the Arduino. Using this method one master can control multiple slaves. There are four wires used in order for transmission of data to take place between devices. Master Output/ Slave Input (MOSI) is the line for the master to send data to the slave. The Master Input/Slave Output (MISO) is used by the slave to send data to the master. The Clock line (SCLK) is the line

used by the clock signal. Finally the Slave Select/Chip Select (SS/CS) line allows the master to select which slave to send data to.

The clock signal synchronises the output of data bits from the master to the sampling of bits by the slave. One bit of data is transferred in each clock cycle. Communication is always initiated by the master since the master generates and configures the clocks signal. The master can choose which slave it wants to communicate with by setting the slave's SS/CS line to a low voltage level. When there is no transmission of data the slave's SS/CS line is set to a high voltage level. The master sends data bit by bit to the slave through the MOSI line and the slave receives this data at the MOSI pin. The slave can also send data back through the MISO line.
The advantages of using this method include:
- No start and stop bits so data can be continuously streamed without interruption
- No complicated slave addressing system as with I2C
- The MISO and MOSI lines are separate therefore data can be sent and received at the same time

The disadvantages of this method are:
- Uses four wires compared to just the two when using I2C
- There is no way of telling the data has been received successfully as there are no ACK/NACK bits as with I2C
- Only allows for a single master

3.4.2.Inter-Integrated Circuit (I2C)

The I2C bus is a bi-directional dual-wire serial bus that supplies a communication link between integrated circuits **(Jung, F. 2005)**. It allows for multiple slaves to be connected to one master like with SPI **(Circuit Basics B. 2016)**. I2C only uses two wires for transmission of data between devices. One of these wires is called Serial Data (SDA) which is used by both devices master and slave to send and receive data. The other is called Serial Clock (SCL) which is the wire that carries the clock signal. It is a serial communication protocol so data is transferred bit by bit along the SDA wire. With I2C data is transferred in 'messages'. Messages are then further broken into frames of data. Each message includes:
- An Address Frame - A 7 to 10 bit unique sequence to each slave that identifies the slave when the master wants to talk to it.
- One or more Data Frames - made up of the data being transmitted.
- Start Condition - The SDA line switches from a high voltage level to a low voltage level before the SCL line switches from high to low.
- Stop Condition - The SDA line switches from low voltage level to a high voltage level after SCL line switches from low to high.
- Read/Write Bit - A single byte specifying whether the master is sending data to the slave (low voltage level) or requesting data from it (high voltage level).
- ACK/NACK Bit - Each frame in a message is followed by and acknowledge/no-acknowledge bit. If an address frame or data frame is successfully received, an ACK bit is sent from receiving device to the sending device.

The advantages of using this method include:
- Only uses two wires
- Supports multiple masters
- ACK/NACK bit gives confirmation that each frame is transmitted successfully
- Well know and widely used protocol

The disadvantages of using this method are:

- Slower data rate than SPI
- The size of the data frame is limited to 8 bits
- More complicated hardware required compared to SPI

3.4.3.Serial Communication

Transistor-Transistor Logic (TTL) Serial and USB Serial are the easiest to set up but not the fastest in terms of data transfer speeds. TTL differs from USB simply in the fact that the connection would be made using a handful of jumper wires instead of a single USB cable. Serial communication has the advantage that data can be sent from either side whenever the need arises **(Reddit. 2014)**.

3.4.4.Conclusion

I decided to use serial communication as it was the easiest to set up and as there was no need for high speed communication between the two devices. Furthermore this method of communication allows for either side to send data when the need arises whereas the SPI and I2C both require the master to initiate communication There are ways in which this problem can be overcome with some extra wiring and code. The only reason for going with TTL serial communication would be because of a lack of USB ports on the Raspberry Pi but as it comes with four inbuilt ports this is not a problem.

3.5. Sensors

3.5.1.Ultrasonic Distance Sensor

Ultrasonic sensors measure distance from the sensor to the target object by using electrical-mechanical energy transformation **(Sharma, A)**. For this project I used the HC-SR04 ultrasonic distance sensor, this sensor makes use of sonar waves to determine the distance to the target object. It consists of a transmitter and a receiver module. The reason for choosing this sensor is that it is relatively cheap hence keeps with the aim of creating a less expensive robot than whats already on the market. Furthermore, it is extremely easy to use and it gives stable and accurate readings. It can give accurate readings ranging from two centimetres unto four hundred centimetres. Moreover, it will still perform accurately even when exposed to sunlight. Its only short coming is that it finds it difficult to detect softer materials such as cloth **(Random Nerd Tutorials)**. Below you can see what the HC-SR04 sensor looks like:

FIGURE 3: HC-SR04 — http://microcontrollerelectronics.com/distance-sensing/

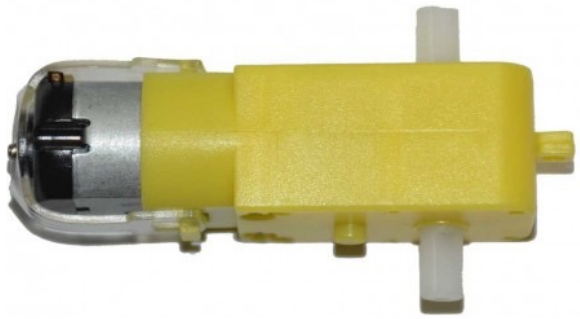
3.6. Motors

Motors play an important roll in robotics. There are different types of motors available each of which are suited for different applications but work on the same underlying principle.

3.6.1.DC Motors

DC (Direct Current) motors have two wires (power and ground) and they continuously rotate. Which means that when you supply the motor with power it will continuously spin until you cut off the power supply. DC motors can run at relatively high revolutions per minute (RPM). The speed of the motor can be controlled using pulse width modulation (PWM), which involves rapidly pulsing the power on and off **(Mod My Pi. 2013)**. Below you can see what a DC motor might look like:

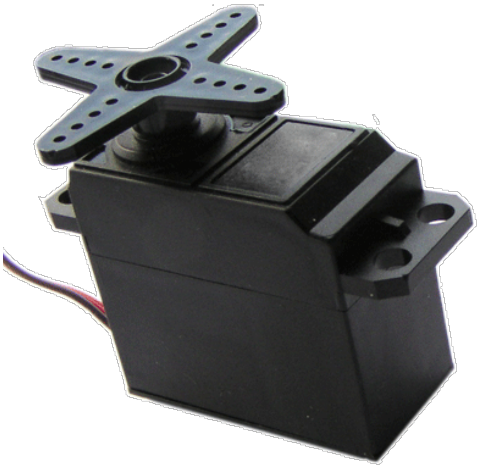
FIGURE 4: DC MOTOR — http://proveedor.com.pk/DC-Motor-Gear-Wheel



3.6.2.Servo Motors

Next, we have servo motors, whose position can be controlled with greater accuracy compared to standard DC motors, therefore, they are used for tasks where the positioning is fundamental and where it needs to be defined accurately for example moving a robotic arm within a given range. Servo motors are assembled from four things: a DC motor, a gearing set, an integrated circuit and a position sensor. Unlike DC motors, servo motors do not have the ability to rotate freely, instead, they are limited to rotate to around 180 degrees. When the motor is instructed to move, it moves to the position and then holds that position, it will resist being moved from that position **(Mod My Pi. 2013)**. Below you can see what a servo motor might look like:

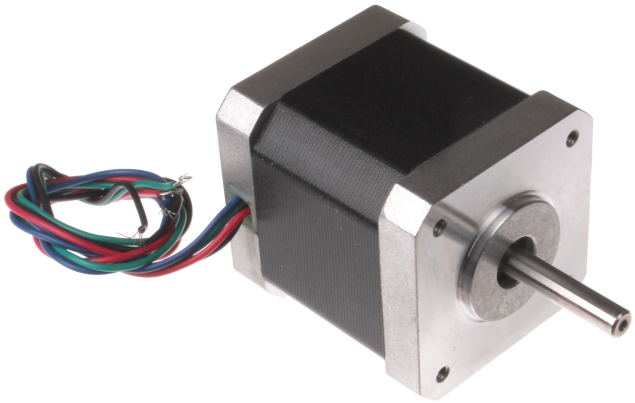
FIGURE 5: SERVO MOTOR — http://stm32f4-discovery.com/2014/10/library-42-control-rc-servo-stm32f4/



3.6.3.Stepper Motors

Lastly, stepper motors, which are in essence servo motors but are motorised using a different method. They use electromagnets to define position instead of a DC motor and an integrated circuit controller **(Mod My Pi. 2013)**. Below you can see what a stepper motor might look like:

FIGURE 6: STEPPER MOTOR — HTTPS://SIMPLE.WIKIPEDIA.ORG/WIKI/STEPPER_MOTOR



3.6.4.Conclusion

DC motors provide a fast, continuous rotation and have a high RPM. Servo motors are also fast, but they do not provide continuous rotation and only provide rotation within a certain range. Therefore, they were unusable to rotate the wheels a full three hundred and sixty degrees. The stepper motors are slower but have the advantage of position control which would make it easier to instruct the robot to move over a certain distance. I chose to use DC motors as stepper motors would require more power and complicated setup compared to the DC motors and the issue of moving the robot a certain distance can be solved using distance is equal to speed times time.

3.7. Chassis

The robot would need a chassis to sit on, this chassis would need to be sourced easily and at an affordable price. Furthermore it had to be large enough to hold all the components. When searching for a chassis that met this criteria there were many options available however a large number of them being too costly or being shipped from too far away. Therefore the decision was made to go for a four wheel based chassis with two decks meaning there would be enough space to host the components. Furthermore this particular chassis was being shipped with DC motors which were the motors I had decided to use.

4. SPECIFICATION, DESIGN and TESTING

4.1. Introduction

As a result of the research carried out on the creating a robot using a Raspberry Pi and an Arduino, the scope of the project was narrowed down to an Arduino vehicle-based robot that would have the ability to avoid obstacles and time permitting the ability to follow a line. The main constraint or limitation of this project was the lack of articles and papers published regarding creating a robot vehicle using an Arduino and a Raspberry Pi. This is because often there is no need for the both as the either is capable on its own, the reason I used the Arduino was due to the fact that it is a micro-controller and not a computer, therefore, it is more suited to embedded projects such as this one compared to the Raspberry Pi. The advantage of using the Arduino is that it is smaller in size and lighter than the Raspberry Pi and in the future there would be potential to add more functionality to the robot by using various types of analogue sensors which the Raspberry Pi is not compatible with straight out of the box. Furthermore, the increased amount of complexity that rose from using both the Arduino and the Raspberry Pi in unison meant that it would be extremely likely that the project would require more time than that allocated. Moreover, the main aim of this project is to create an outreach engagement for students ranging from fourteen to fifteen years using the robot as a tool, meaning that it needed to be functional in order for the primary outreach activity to take place.

Once the research was completed and the general definition of the product was defined the next step was to create a concept design and create a project plan in order to achieve this concept within the given time period. There are two parts to this project the first is creating the robot and the second is using that robot to teach school children the fundamentals of programming. I will start with discussing the specification and design of the robot then move on to talk about the teaching and learning materials that are to be used with the robot.

4.2. Robot

4.2.1.Iteration 1

4.2.1.1. Requirements

- The robot should be able to move forwards and backwards
- The robot should be able to rotate in either direction left or right
- The robot should be able to detect obstacles using an ultrasonic distance sensor

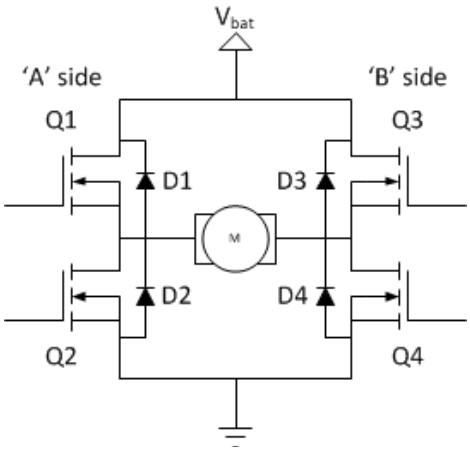
4.2.1.2. Design

A DC motor converts electrical energy to mechanical energy. Therefore, they need to be powered by an electrical source. When the terminals are connected the motor spins in one direction. If you switch the connections around the motor will spin in the opposite direction. This solves the problem of making the motors spin in both a forward and reverse direction. However, it would be impossible to physically switch the connections around each time the motor is required to operate in the other

direction. Therefore, the motor is connected to the Arduino via an h-bridge, specifically the L293D. This is a typical widely used and highly affordable (£1) motor controller which will allow the DC motors to drive the wheels in both directions, i.e. forwards and backwards without the need to switch the connection around on the motor. It is widely used in robotics mainly due to its affordability and size.

A H-Bridge is a circuit which allows the voltage to flow in either direction **(RakeshRon. A. 2013)**. Below is a schematic of single H-bridge:

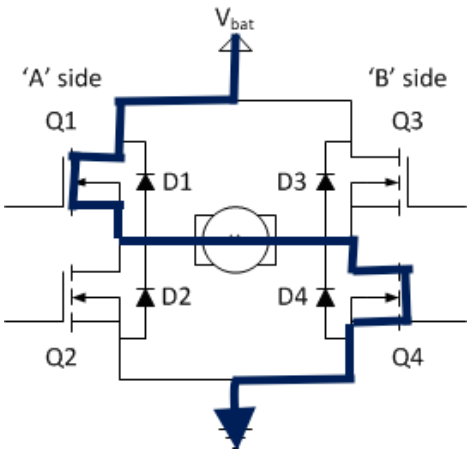
FIGURE 7: H-BRIDGE CIRCUIT — http://www.modularcircuits.com/blog/articles/h-bridge-secrets/h-bridges-the-basics/



There are four switching elements (Q1..Q4) and there are four diodes (D1..D4) **(Modular Circuits)**. The power is supplied from the top of the bridge (Vbat) and the bottom of the bridge is connected to ground. The centre labeled M is the load which in this case is a DC motor.

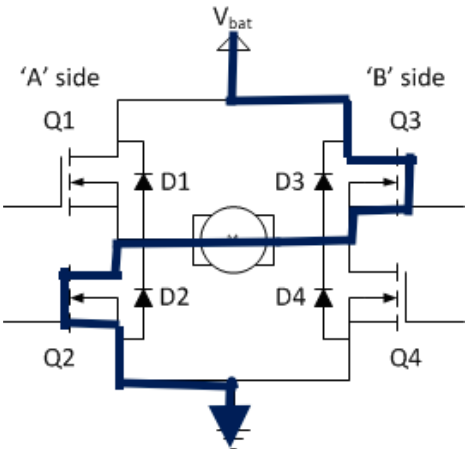
With the DC motor connected as the load if Q1 and Q4 are switched on the left-hand side of the current will flow through the motor from the left-hand side. This is because the left lead of the motor is connected to the power and the right lead to the ground. Therefore, the current will pass through the motor, powering it up, as shown in the circuit below:

FIGURE 8: H-BRIDGE CIRCUIT CURRENT FLOW FORWARDS DIRECTION



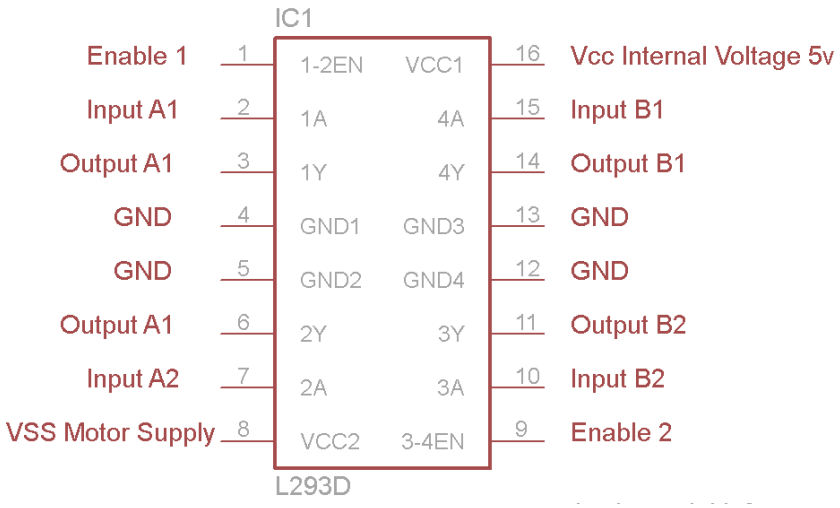
This would power the motors in one direction lets say for example this is the forwards direction. In order for the bridge to power, the motors in the reverse direction the switching elements Q2 and Q3 would need to be turned on. This would let the current flow through the motor from right to left as shown below:

FIGURE 9: H-BRIDGE CIRCUIT CURRENT FLOW BACKWARDS DIRECTION



The L293D consists of two of these circuits, therefore, it is labeled as a dual h-bridge. Giving it the capacity to power two DC motors separately. The figure below shows the pin diagram of the L293D:

FIGURE 10: H-BRIDGE PIN DIAGRAM — http://www.rakeshmondal.info/L293D-Motor-Driver



There is one h-bridge on the left-hand side and another on the right. Hence, there are two enable pins 1 and 9, on either side. These pins need to be set to high (digital value) in order to drive the motor. If the motor is being driven from the left-hand side then the value of pin 1 needs to be set to high and the same goes for the right-hand side except it would be pin 9 being set to high. If either of these pins were to be set to a low value then the motor connected to that side would simply stop spinning i.e. come to a stop. As the figure shows, there are two input pins and two output pins on either side. The output pins are connected to the terminals of the motor and the input pins to the Arduino. The motor connected on the left-hand side will be controlled from pins two (InputA1) and seven (InputA2), the direction in which the motor will rotate depends on the logic inputs provided to the two pins **(RakeshRon. A. 2013)**.

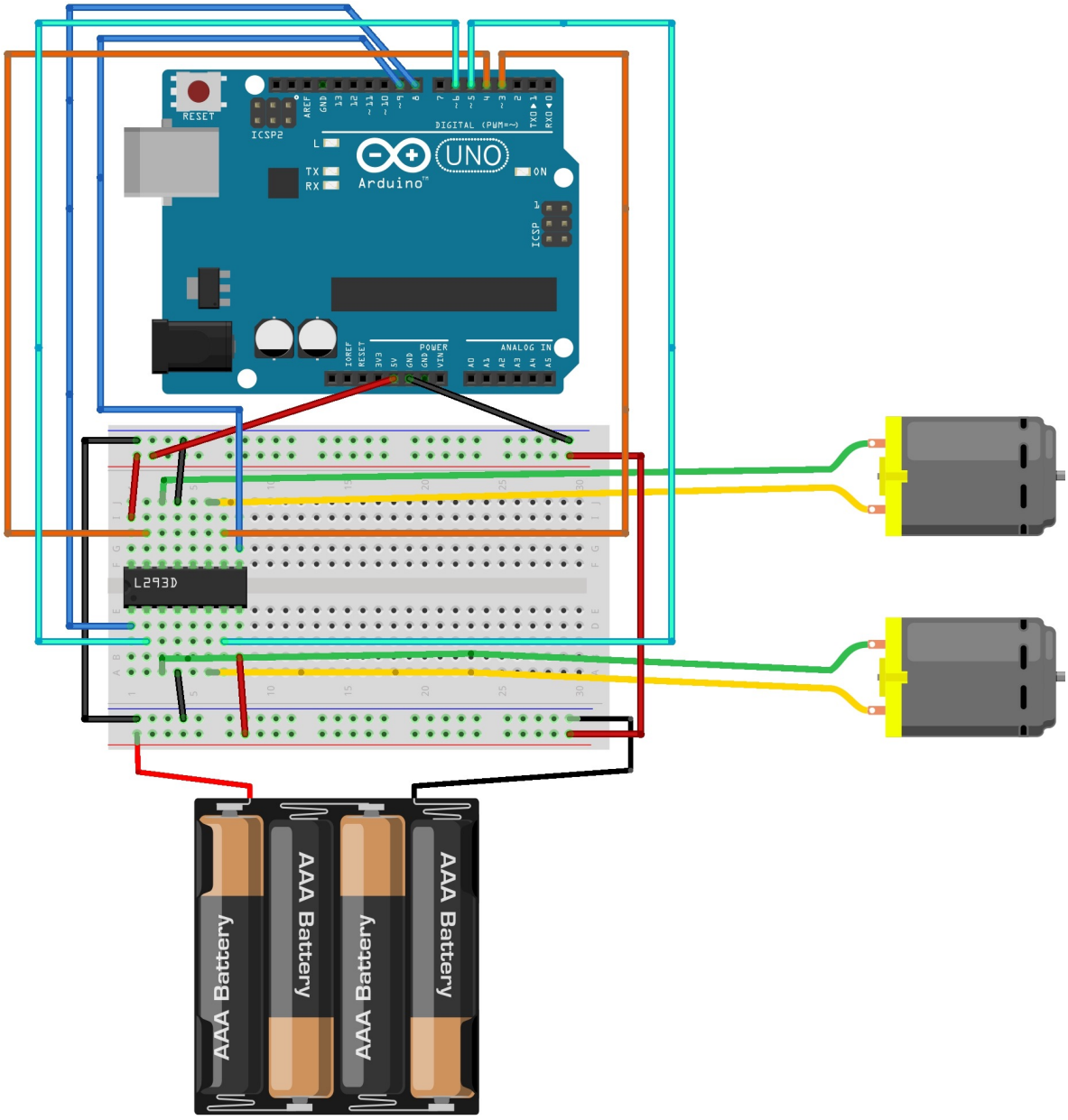
The input can either be 1 or 0 (high or low). Below is a logic table for powering a motor which is connected across the left-hand side to output pins three and six:

	Pin 2	Pin 7
Clockwise	1	0
Anti-clockwise	0	1
Idle	0	0
Idle	1	1

Figure 11 displays a schematic that shows how to implement two DC motors to an Arduino via a dual h-bridge. The red and black wires represent positive wires and negative (ground) wires respectively. The battery pack is connected to the breadboard via the positive and negative rails on the bottom side of the board. The same goes for connecting the Arduino to the board, a wire is led from the five volt output pin to the positive rail and from the ground pin to the negative rail. Both enable pins on the L293D are connected to the Arduino via pins eight and nine via the blue wires. The input for the top motor comes from pins three and four on the Arduino which are connected via the orange wires to pins nine and fifteen on the L293D. Whereas the input for the bottom motor is represented by the two cyan coloured wires. Lastly, the motors are connected to their relevant output pins on the L293D. The battery pack is needed as the Arduino does not have enough power to power the motors on its own and if this is done it can lead to too much strain on the Arduino hence risking breaking it. Each battery supplies one and a half volts, therefore, giving us a total of six volts from the battery pack. The motors have a rating from three to six volts which means that they can run at any voltage between three and six volts. Therefore, each motor will receive its minimum three volts of power in order to spin the motor. However if they were to receive a full six volts each then the torque produced by the motor would be greater, hence, spinning the wheels a lot faster.

FIGURE 11: SCHEMATIC FOR CONNECTING DC MOTORS TO ARDUINO VIA H-BRIDGE

The robot chassis was provided with four motors and four wheels but there is the problem of powering all four motors, as using a single battery pack of four AA batteries will mean that the motors only receive one and a half volts of power supply instead of the minimum voltage of three volts that they should be supplied with. Therefore theoretically it should not be possible to power all four motors using four AA batteries.



Schematic made with Fritzing

4.2.1.3. Test - Powering Two or Four Motors

When carrying out tests to see if four AA batteries could power all the motors it was evident that the batteries were, in fact, capable of powering all four motors however as the power is divided between four motors, the motors spin slower compared to when only two motors are powered by the same power source. In order to prove this, a small test was devised in which four motors would be hooked up to a breadboard which was powered by the four AA batteries (shown in figure 14) and then the sound of one of the motors spinning would be recorded. Next, two motors are disconnected from the breadboard leaving just two motors powered by the four AA batteries and the same test is conducted. The next step was to look at the sound waves from the recordings. If the motor is spinning at a higher rpm then the waves should reach a higher peak value. The following figures show the sound waves of when four motors were running and when only two were powered respectively.

FIGURE 12: SOUND WAVE OF THE MOTOR RECEIVING 1/4 OF THE POWER

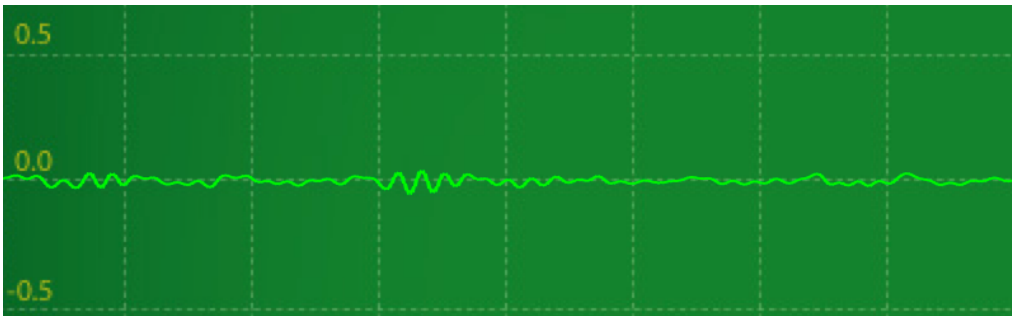
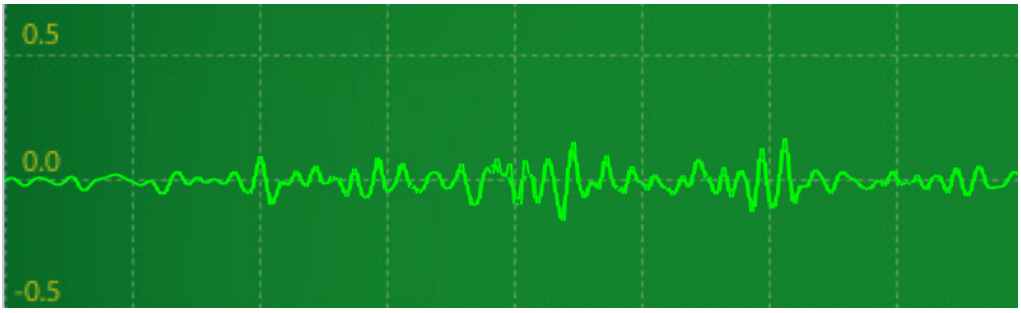


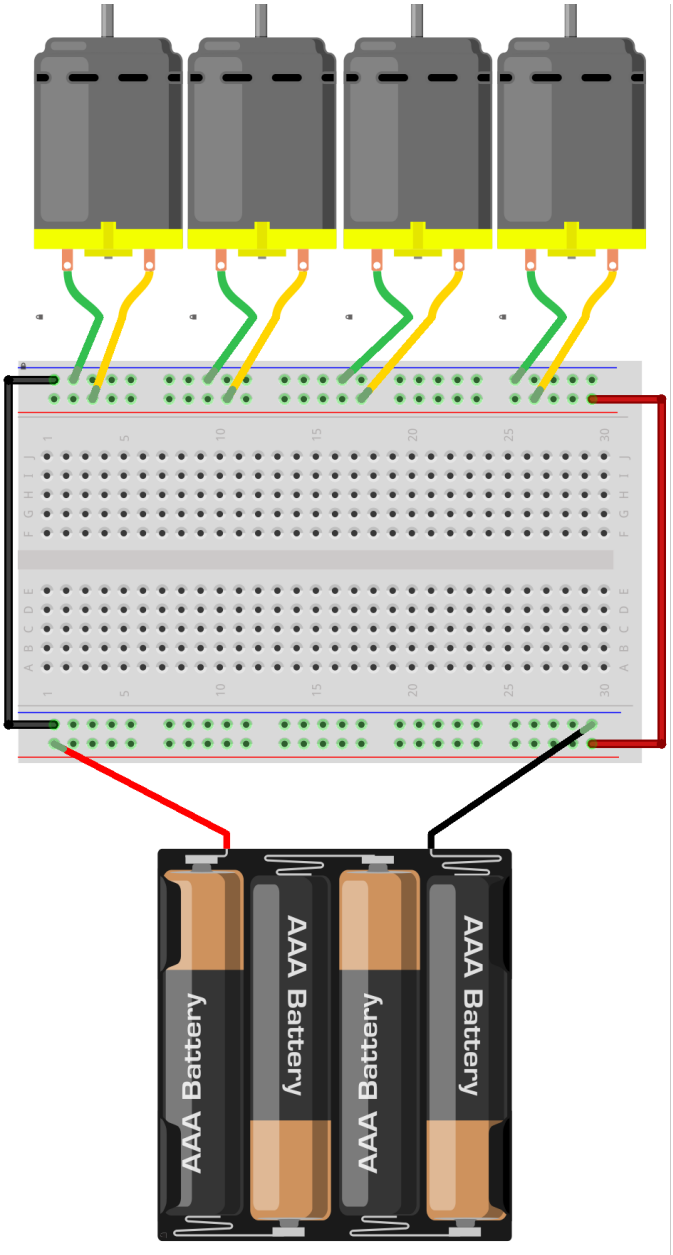
FIGURE 13: SOUND WAVE OF THE MOTOR RECEIVING 1/2 OF THE POWER



Due to the lack of power when using all four motors I decided that it would be best for me to only power the back two motors. Unfortunately due to the resistance from the two wheels that were no longer powered by the motors the robot was unable to move in either direction.

In order to solve this problem the two motors that were not being powered were taken apart carefully and opened up to remove the motor heads, which lead to the wheels spinning more freely. The batteries were hooked up again but the resistance of the two unpowered wheels was still too great for the total torque produced by the two rear motors. Hence, the only solution was to completely open up the motors and remove the cogs that were grinding against each other causing the friction. After the removal of these cogs, the robot was, in fact, able to move in the forwards and backwards direction. These tests were all carried out without the Arduino as there was no need for it, instead, the motors were simply connected directly to the power source as shown in figure 14 below.

FIGURE 14: SCHEMATIC FOR POWERING MOTORS DIRECTLY FROM POWER SUPPLY



The next step was to connect the motors to the Arduino via the H-bridge as shown in figure 11 and create a simple program that would get the robot to move in the forwards direction for a certain amount of time and then in the backwards direction. I used an open source Arduino library called DCMotorBot which is created by Sudar Muthu **(Muthu, S.)**. The library allowed for control of two DC motors connected to the Arduino via a h-bridge. Below in figure 15 you can see some example code that I used in order to make the robot move in the forwards direction for 5 seconds and then in the reverse direction for another 5 seconds.

FIGURE 15: CODE USING DCMOTORBOT LIBRARY

```
#include <DCMotorBot.h>

DCMotorBot robot;

void setup() {
    // initialize bot pins
    bot.setEnablePins(8, 9);
    bot.setControlPins(5, 6, 3, 4);
}

void loop() {
    //Get the robot to move forward for 5 seconds
    robot.moveForward();
    delay(5000);
    //Get the robot to move backwards for 5 seconds
    robot.moveBackward();
    delay(5000);
}
```

Now that the robot is able to move in the forwards and backwards direction, the next step is to get it to rotate in either direction. In order to rotate either left or right, the robot would have to use a method known as differential steering **(RakesnRon. B. 2014)**. Which is simply creating a difference in the speed of two adjacent wheels on the robot. In this case, the rear left and right wheels. For the robot to perform a three hundred and sixty-degree turn on the spot the left and right wheels would have to spin in opposite directions at the same speed. The DCMotorBot library had inbuilt functions for rotating the robot in both directions, that applied this exact principle. Therefore, the code below in figure 16 was used to test if the robot was, in fact, able to rotate in the left direction.

FIGURE 16: CODE USED TO ROTATE ROBOT IN LEFT DIRECTION

```
#include <DCMotorBot.h>

DCMotorBot robot;

void setup() {

    // initialize bot pins
    bot.setEnablePins(8, 9);
    bot.setControlPins(5, 6, 3, 4);
}

void loop() {
    robot.turnLeft();
    delay(2000);
}
```

However when trying this method a problem arose, in order for the robot to rotate left the rear right wheel needed to spin in the forwards direction and the rear left wheel in the backwards direction. The high traction from the front two wheels caused an opposing force making it difficult for the robot to rotate, furthermore, the motors were only receiving their minimal voltage supply (three volts) hence were producing less torque than they were capable of, meaning the robot did not have enough power to pivot on the spot and simply remained stationary.

If more batteries were installed in order to get the additional six volts of power to power the two motors at full capacity then there would be an increase in the total amount of weight on the chassis, which would, in turn, lead to a greater amount of traction for all four wheels due to the increased downforce created. Which means that adding more power would lead to increased weight. Furthermore adding more power would mean needing more batteries which are costly. The project already uses four AA batteries in order to get the additional power needed the number of batteries would need to be doubled to eight, increasing the cost of running the robot by nearly double. The only solution to this would be to decrease the distance apart the two front wheels are from the back two **(RakeshRon. B. 2014.)**. This would lead to a less opposing force created by the front wheels allowing the rear two wheels to rotate the robot on the spot.

Therefore, I decided to scrap the idea of only powering two motors and go back to powering all four motors. The two motors that were taken apart earlier were reassembled in order for them to operate like they should rather than spin freely without any resistance. The motors were reattached to the chassis and the same logic was applied as with the two wheels but now with all four wheels being powered. So to turn in the left direction the two motors on the right-hand side were powered to spin forwards at full speed and the two left motors to spin in the reverse direction at the same speed.

The DCMotorBot library is built to only work with two DC motors therefore the code for the library needed to be changed in order to add functionality to control all four DC motors. Once the library was modified the robot still remained stationary and did not rotate in the left direction on its axis. In order to make sure that it wasn't the lack of power causing this, the circuit was powered up using the mains power supply and even though the motors visibly and audibly had more power, the robot would still not perform a full rotation on its axis. It would manage to rotate roughly 30 degrees but then due to the resisting force and traction in the wheels, it would come to a halt.

Therefore I came to the conclusion that only way for me to get the robot to turn would be to modify the chassis. As I did not have the tools nor the skills to use the tools to take this approach I decided to look for a new chassis. In terms of four wheel based chassis, there was not much available with the shorter wheelbase needed that would be delivered in time. Therefore, I opted to go for a two-wheeled approach and ordered a new two-wheel based chassis.

The two wheel chassis arrived a few days later and was assembled with relative ease compared to the four wheel chassis because it came with an instruction manual which showed how to assemble the chassis step by step. This just went further to prove that the four wheel chassis ordered previously was of poor quality as it was extremely tedious to assemble and also did not come with instructions on how to assemble it. Once the chassis was assembled the two motors were connected as shown in figure 11. I tested the motors with the code below in figure 17 and they worked perfectly fine the robot was able to move in the forwards and backwards direction and it had the ability to rotate in either the left or right direction on the spot.

FIGURE 17: CODE TO TEST TWO-WHEEL BASED ROBOT

```
#include <DCMotorBot.h>

DCMotorBot robot;

void setup() {
    // initialize bot pins
    bot.setEnablePins(8, 9);
    bot.setControlPins(5, 6, 3, 4);
}

void loop() {
    robot.moveForward();
    delay(2000);
    robot.moveBackward();
    delay(2000);
    robot.turnRight();
    delay(2000);
    robot.turnLeft();
    delay(2000);
}
```

4.2.1.4. Test - Iteration 1

Test Case ID: 1	**Test Purpose:** To ensure the robot can move in the forwards and backwards direction, and can rotate on the spot in either direction.		
Environment: The motors are connected to the robot using a h-bridge and the sketch must have been created.			
Preconditions: Arduino must be connected to the motors. The motors must be powered by 4xAA battery pack and the Arduino must be powered by 9V battery. The sketch must be bug free.			
Test Case Steps:			
Step No	**Procedure**	**Response**	**Pass/Fail**
1	User loads the sketch in Figure 17.	User is taken to the sketch window.	Pass
2	User uploads sketch onto Arduino	Message saying upload is complete	Pass
3	Switch on Battery Pack	The robot should move in the forwards direction for 2 seconds then in the backwards direction for 2 seconds. Followed by rotating left for 2 seconds and then rotating right for 2 seconds.	Pass
Comments: The robot runs successfully and moves in forwards and backwards direction, and rotates in either direction			
Author: Shanay Shah		**Checker:** Shanay Shah	

4.2.1.Iteration 2

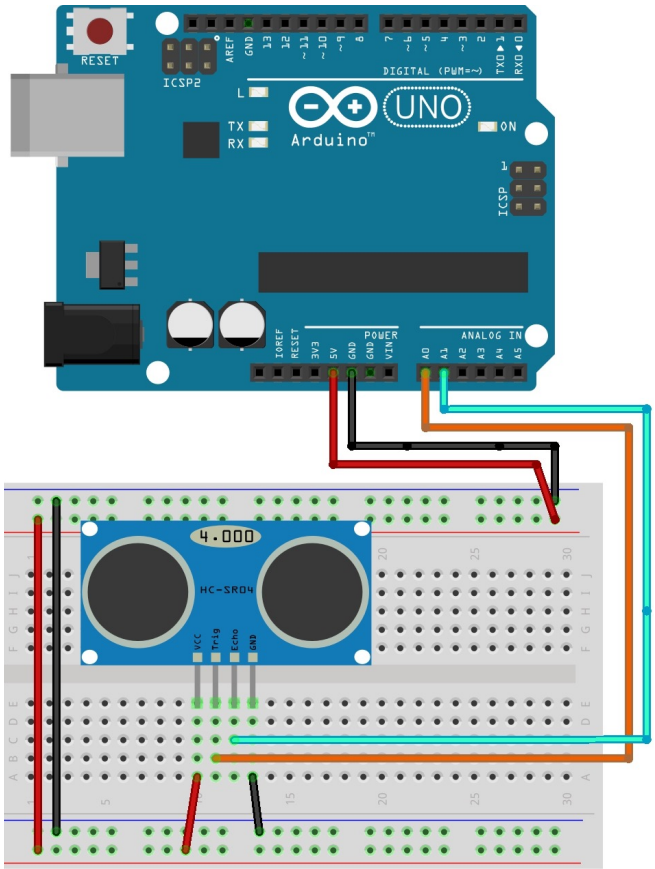
4.2.1.1. Requirements

- The robot should be able to detect obstacles using an ultrasonic distance sensor and come to a stop

4.2.1.2. Design

Now that the robot is programmed and wired up to either move in the forwards, backwards, left or right direction, the next step is to get the robot to read obstacles when they are a certain distance away and stop the motors in order to get the robot to come to a halt before coming in contact with the obstacle. For the robot to have this functionality it must employ the use of an ultrasonic distance sensor. Specifically, as mentioned earlier the HC-SR04. The schematic below shows how the sensor is connected to the Arduino:

FIGURE 18: HCSR04 TO ARDUINO CONNECTION



The first step is to attach the HC-SR04 to any four consecutive pins on the breadboard. The HC-SR04 has a VCC pin which is an input pin for power of up to five volts connected to the positive rail of the breadboard and a ground pin which connects to the negative rail of the breadboard. The sensor also has input and output pins named trigger and echo pins respectively. The trigger pin is connected to analog pin A0 on the Arduino whereas the echo pin is connected to A1. It works by sending out a pulse of sound which is controlled from the trigger pin and then reading how the pulse was reflected back through the echo pin on the sensor. The breadboard needs a source of power hence the five volt output from the Arduino is connected to the positive rail of the breadboard and the ground pin is connected to the negative rail. Both the top and the bottom of the breadboard need to receive power and be grounded hence you run a power cable from the top positive rail to the bottom positive and rail and the same goes for the ground wire except it will link up the two negative rails.

4.2.1.3. Test - Distance Sensor

In order to test if the distance sensor was working, I created a circuit like the one above except I attached an LED to the circuit. The purpose of the LED is to switch on every time the ultrasonic distance sensor picked up an object in front of it within the pre-defined range (safe zone). Below you can see the schematic (figure 19) and the code (figure 20) for this test.

FIGURE 19: SCHEMATIC FOR TEST OF HC-SR04

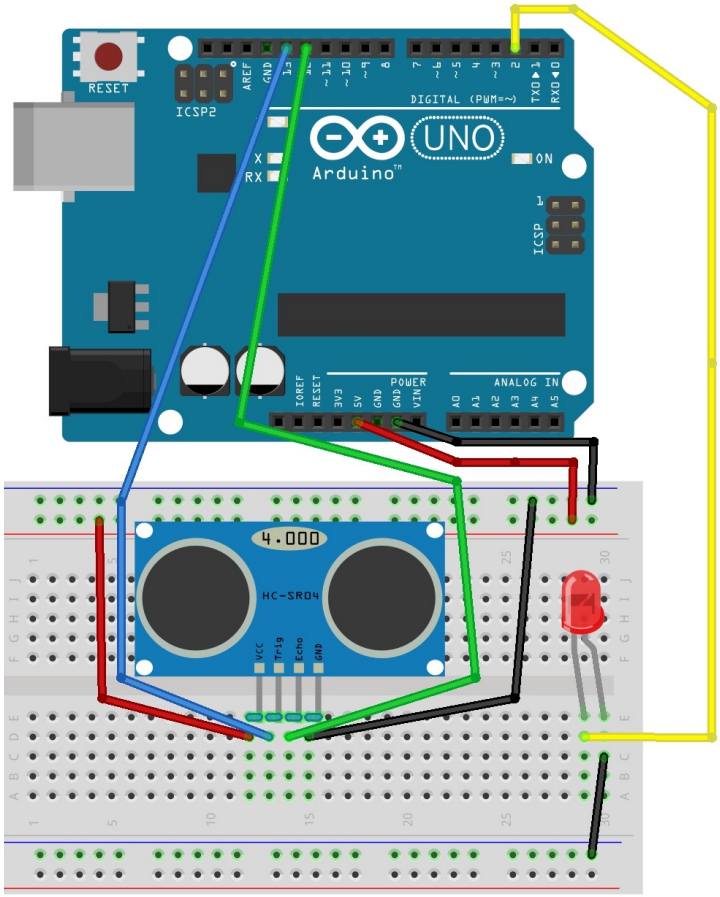


FIGURE 20: CODE FOR TEST OF HC-SR04

```
#include <NewPing.h>
int trigPin = 14;
int echoPin = 5;
int maxDistance = 200;
int safeZone = 5;
int redLed = 2;
NewPing sonar(trigPin, echoPin, maxDistance);

void setup()
{
  Serial.begin(9600);
}

void loop()
{
  pinMode(redLed, OUTPUT);
  unsigned int distance = sonar.ping_cm();
  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println("cm");
  if (distance > safeZone)
  {;
    digitalWrite(redLed, LOW);
  }
  else
  {
    digitalWrite(redLed, HIGH);
  }
  delay(1000);
}
```

There are many existing default libraries for many Arduino compatible devices such as stepper motors and WiFi modules, however, none exist for distance sensors. Whilst going through tutorials **(Random Nerd Tutorials)** on how to use the HC-SR04 with an Arduino I found the NewPing library **(Eckel, T)**. Therefore, the first line of code is to include this library or in other words to import this library.

The line of the code *NewPing sonar(trigPin, echoPin, maxDistance)* is used to create an object of the NewPing class. In the setup phase of the code, we begin a connection to the serial output of the Arduino so that we can receive output to the serial monitor. Then in the loop section of the code, we tell the Arduino that the pin leading out to the red led is required to output a signal. The next steps is used to get the distance of objects from the sensor. Using the functions built into the NewPing library the distance of the object is relayed back in centimetres. Lastly, there is an if statement which checks whether or not the object is outside of the safe zone and if it is not then it switches on the LED. This sketch produced the following output to the serial monitor when I moved my hand towards the sensor:

FIGURE 21: OUTPUT FROM SERIAL MONITOR FOR TEST OF HC-SR04

```
Distance: 48cm
Distance: 40cm
Distance: 36cm
Distance: 28cm
Distance: 21cm
Distance: 15cm
Distance: 12cm
```

Note that this was in order to test and understand the logic behind how the distance sensor operated, in reality, there would be no need for the LED. instead of switching on an LED when an object enters the safe zone the motors would need to come to a halt, then the robot would have to go in the backwards direction and turn either left or right to avoid the obstacle, once the obstacle was out of the safe zone distance the robot would then move in the forwards direction again. But with this iteration, the aim was to get the robot to detect the obstacle and come to a halt.

The next step was to connect up the motors and ultrasonic distance sensor to the breadboard (shown in figure 24) and then created a sketch which includes a function scan() that scans for obstacles (shown in figure 22):

FIGURE 22: FUNCTION FOR SCANNING FOR OBSTACLES

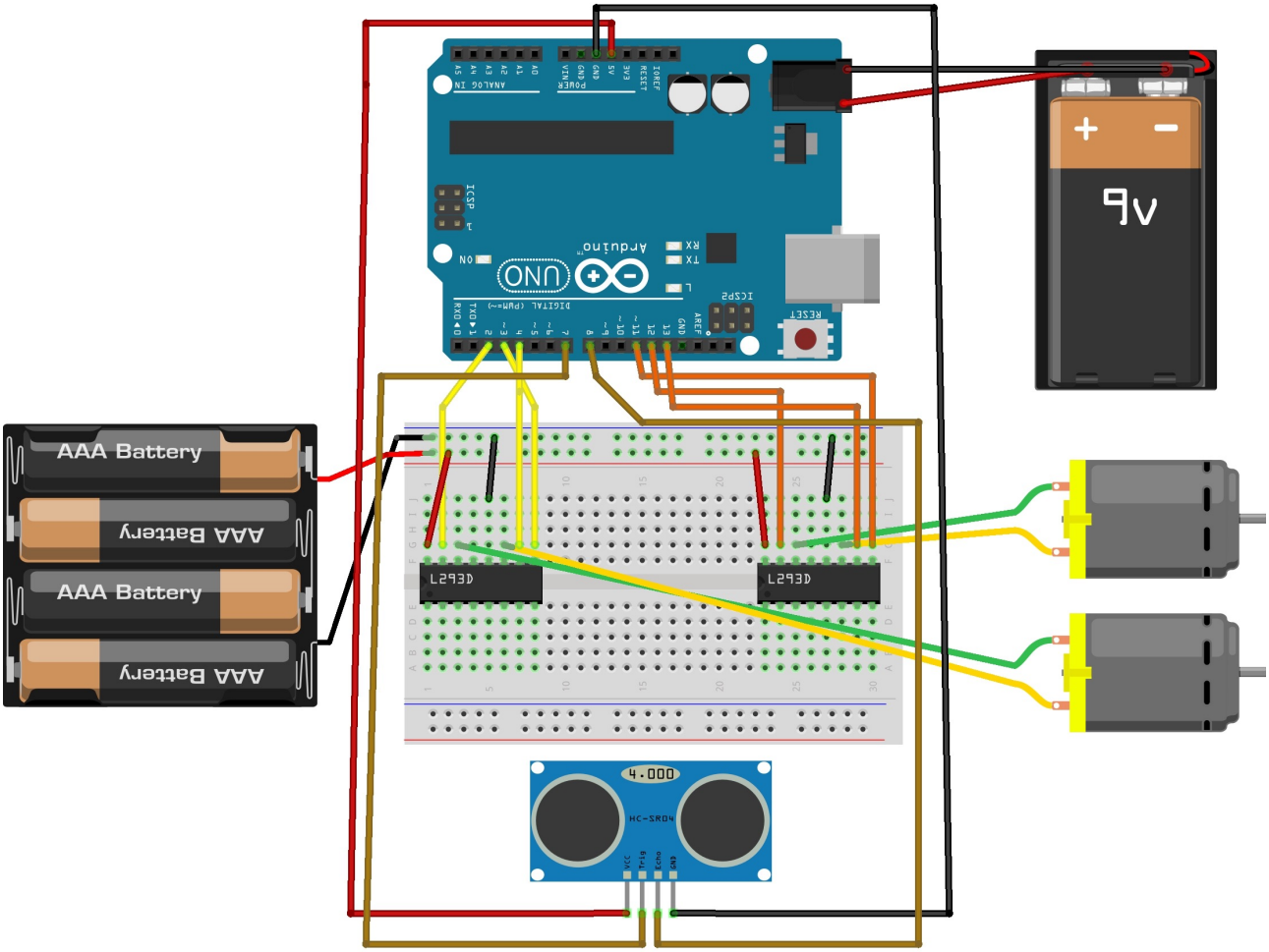
```
void scan()
{
    cm = sonar.ping_cm();
}
```

FIGURE 23: CODE FOR OBSTACLE DETECTION AND STOPPING ROBOT

```
#include <NewPing.h>
#include <DCMotorBot.h>

int triggerPin = 7;
int echoPin = 8;
int maxDistance = 200;
int safeZone = 5;

NewPing sonar(triggerPin, echoPin, maxDistance);
DCMotorBot robot;

//Variable Declarations
unsigned int distance;
unsigned int cm;
void setup()
{
  Serial.begin(9600);
  robot.setEnablePins(8, 9);
  robot.setControlPins(5, 6, 3, 4);
}

void loop()
{
  scan();
  distance = cm;
  Serial.println("Front distance = ");
  Serial.print(distance);
  if(distance >= safeZone || distance == 0)
  {
   robot.moveForward();
  }
  else
  {
    robot.stop();
  }
}
```

This code (figure 23) first sets up the various variables and pin numbers and then in the loop section checks to see whether there is anything within the safe zone of the robot and if no object exists within this range then the robot begins to move in the forwards direction. If an object is detected within the safe zone the robot instructs the motors to stop and come to a halt. Before crashing into the obstacle.

FIGURE 24: SCHEMATIC FOR CONNECTING MOTORS AND HC-SR04 TO BREADBOARD



After running the code (figure 23) the robot did not move in the forwards direction to begin with so I checked that the batteries were properly connected to the breadboard and that all the connections were properly made. However even with all the connections in good tact, the robot did not move. The next step was to check the code, even the code checked out and ran without any error. Therefore, I decided to stop using the open source DCMotorBot library to control the motors but instead hard code functions that would give the robot the ability to move in all the directions. The functions created for going forwards, backwards, stopping and rotating left and right can be found in the figure below (figure 25). This was because I could not think of anything else that may be causing the problem. Note even though I had previously modified the library to control four motors I had re-downloaded and installed the library back to its original version before attempting the above.

FIGURE 25: FUNCTIONS CREATED FOR ROBOTS MOVEMENT IN ALL DIRECTIONS

```
void right()
{
  //Serial.println("Turning Right");
  digitalWrite(rightTerminal1, HIGH);
  digitalWrite(rightTerminal2, LOW);
  digitalWrite(leftTerminal1, LOW);
  digitalWrite(leftTerminal2, HIGH);
}

void halt()
{
  //Serial.println("Stopping");
  digitalWrite(rightTerminal1, LOW);
  digitalWrite(rightTerminal2, LOW);
  digitalWrite(leftTerminal1, LOW);
  digitalWrite(leftTerminal2, LOW);
}
void forward()
{
  //Serial.println("");
  //Serial.println("Moving Forward");
  digitalWrite(rightTerminal1, LOW);
  digitalWrite(rightTerminal2, HIGH);
  digitalWrite(leftTerminal1, LOW);
  digitalWrite(leftTerminal2, HIGH);
}

void backward()
{
  //Serial.println("Moving Backward");
  digitalWrite(rightTerminal1, HIGH);
  digitalWrite(rightTerminal2, LOW);
  digitalWrite(leftTerminal1, HIGH);
  digitalWrite(leftTerminal2, LOW);
}

void left()
{
  //Serial.println("Turning Left");
  digitalWrite(rightTerminal1, LOW);
  digitalWrite(rightTerminal2, HIGH);
  digitalWrite(leftTerminal1, HIGH);
  digitalWrite(leftTerminal2, LOW);
}
```

The next step involved testing the robot with these functions instead of the DCMotorBot libraries inbuilt functions and the safe zone set to 10cm and the motors spinning at full speed, the robot did not have enough time to detect the object and bring the motors to a dead stop. Therefore, I decided to increase the distance of the safe zone to 25cm and reduced the power of the motors to 180 (255 being full power), in order for the robot to have enough time to process that there was an obstacle in its path, this resulted in the robot having enough time to detect the obstacle and come to a halt before crashing into it. Below you can see the code using the functions I created for the robots movement.

FIGURE 26: CODE USING SELF MADE FUNCTIONS FOR ROBOT MOVEMENT

```
void loop()
{
  scan();
  distance = cm;
  Serial.println("Front distance = ");
  Serial.print(distance);
  if(distance >= safeZone || distance == 0)
  {
   forward();
  }
  else
  {
    halt();
  }
}
```

However after leaving the robot to run, whilst analysing its behaviour I realised that the robot would not always operate properly and sometimes would come to a halt without there being an obstacle in front of it. The robot was then connected to the computer and run in tethered mode so that I could see the output of the ultrasonic distance sensor as the robot moved around the environment. As I analysed the data being output to the serial monitor I came to realise that often there would be an out of the normal value being returned for the distance. When this distance value was less than the defined safe zone the robot simply came to a halt even though there was no obstacle within the safe zone.

As mentioned in the research review section the ultrasonic distance sensor works by sending out a ping and then waiting for the ping to be bounced back. Leaving me to think that sometimes the ping is being bounced onto a surface that absorbs the sound or reflects it back with greater strength fooling the sensor to think that there is no object nearby or that the object is closer than in actual fact. Therefore, I needed to create some sort of average reading to counteract these outlying values that were being returned in certain cases. The NewPing library comes with a function that calculates the distance relayed back in microseconds as an average value of five iterations of checking the distance. Therefore, I decided to use this function instead of the *ping_cm()* function used above.

Below in figure 27 you can see the code used to implement this new averaging function for the HC-SR04. This code uses the *ping_median()* function which returns a value in microseconds. This value would be of no use to us in this format, therefore, the *convert_cm()* function that comes with the NewPing library is used to convert this microsecond value into centimetres.

FIGURE 27: NEW AND REVISED SCAN FUNCTION

```
void scan()
{
    msTime = sonar.ping_median();
    cmIter = sonar.convert_cm(msTime);
}
```

With the use of this function the robot operated seamlessly and did not come to a halt when there was no obstacle within the preset safe zone.

4.2.1.4. Test - Iteration 2

Test Case ID: 2	**Test Purpose:** To ensure the robot can detect obstacles in its path and come to a stop, avoiding any collision.		
Environment: The motors are connected to the robot using a h-bridge, the HC-SR04 is connected to the Arduino and the sketch has been created.			
Preconditions: Arduino must be connected to the motors. The motors must be powered by 4xAA battery pack and the Arduino must be powered by 9V battery. The sensor must be powered through the 5 volt output pin on the Arduino. The sketch must be bug free.			
Test Case Steps:			
Step No	**Procedure**	**Response**	**Pass/Fail**
1	User loads the sketch in Figure 26 using revised scan function from figure 27.	User is taken to the sketch window.	Pass
2	User uploads sketch onto Arduino.	Message saying upload is complete.	Pass
3	Switch on Battery Pack.	The robot should begin to move forwards and detect an obstacle in front of it and come to a stop.	Pass
Comments: The robot runs successfully came to a stop before colliding with the obstacle ahead of it.			
Author: Shanay Shah		**Checker:** Shanay Shah	

4.2.1.Iteration 3

4.2.1.1. Requirements

- The robot should be able to avoid the obstacles it detects.

4.2.1.2. Design

The robot was now detecting obstacles and not crashing into them by coming to a halt. This does not make the robot useful in any way, the robot needs to realise that there is an obstacle in its path and find a way to navigate around it and carry on. The robot has very simple vision and motion systems, therefore, the obstacle avoidance algorithm will also be relatively simple. When the robot detects and obstacle in its path it will back up in the reverse direction and then rotate in either the left or right direction on the spot until the ultrasonic distance sensor does not pick up an object within the safe zone, at which point the robot will continue to move in the forwards direction. This process will be continuously looped over and over again as long as the Arduino is connected to a power source and switched on.

In order to do this I needed to improve on the code used in the previous iteration, where the robot simply came to a halt when it detected an obstacle, to code that would instruct the robot on how to navigate the obstacle. Therefore I needed to create a function that allowed the robot to choose whether it needed to rotate left or right.

FIGURE 28: NAVIGATION FUNCTION

```
void navigate()
{
  if (random(2) == 0){
    backward();
    delay(200);
    right();
    delay(200);
  }
  else{
    backward();
    delay(200);
    left();
    delay(200);
  }
}
```

As the sensor only detected objects in front of it and had no way of telling whether the object lay to the left or right hand side the code simply chose between the two options using a random() function form the Arduino library. This function takes a range of numbers as its input and returns any one of those numbers randomly as output. In this case, I have used random(2) which either returns 0 or 1. I can then use this result to tell the robot to rotate in either the left or right direction.

The next thing to think about is how long does the robot need to rotate in either direction for, there is no simple answer for this as the robot has no way of knowing how fast it is turning or when it has completed a full turn. In order for the robot to be able to give such feedback, additional hardware such as a compass or magnetometer would be needed. Much like the one that is in the Sense HAT for the Raspberry Pi that allows the Sense HAT to know the orientation of the system. Without any additional hardware I can make the robot turn for a certain number of milli seconds which is hard coded into the code. This is definitely not the most efficient way to do it as implementing delays in the code means that the robot cannot process any information for that period of time. After completion of this phase of the project which also will include the design and implementation of the actual outreach activity which is discussed below and time permitting I will go about trying to implement a more efficient algorithm in which there are no delays and the robots rotation time is not fixed to a certain amount of time. FULL CODE

4.2.1.3. Test - Iteration 3

Test Case ID: 3	**Test Purpose:** To ensure the robot can detect obstacles in its path and avoid them.		
Environment: The motors are connected to the robot using a h-bridge, the HC-SR04 is connected to the Arduino and the sketch has been created.			
Preconditions: Arduino must be connected to the motors. The motors must be powered by 4xAA battery pack and the Arduino must be powered by 9V battery. The sensor must be powered through the 5 volt output pin on the Arduino. The sketch must be bug free.			
Test Case Steps:			
Step No	**Procedure**	**Response**	**Pass/Fail**
1	User loads the final sketch.	User is taken to the sketch window.	Pass
2	User uploads sketch onto Arduino.	Message saying upload is complete	Pass
3	Switch on Battery Pack.	The robot should begin to move forwards and detect an obstacle in front of it and navigate around the obstacle.	Pass
Comments: The robot runs successfully navigates around the obstacles it detects.			
Author: Shanay Shah		**Checker:** Shanay Shah	

4.3. Engagement Activity

Due to the not reaching a high level of functionality and only being able to carry out simple missions, it was decided that it would not be used for the outreach activity. Instead, the outreach engagement activity will involve the students using a Raspberry Pi with the additional Sense HAT add-on to interact with the environment around them as they try and learn some of the basics of computer programming. You can find the worksheet in appendix 8.2. and the fact sheet that goes along with it in appendix 8.3.. This activity has been adapted from the original Getting Started With The Sense HAT learning resource available online **(Raspberry Pi Learning Resources B)**. I decided to include a fact sheet as some of the teachers may not be computer scientists and therefore they are able to refer to the fact sheet when teaching the students, if they are unclear about any of the terminology involved in the worksheet. Furthermore, the teachers section at the beginning of the worksheet gives the teacher an overview of the activity and gives the teacher information on what they will be teaching and what tools they will be using to teach. Moreover, as some teachers may lack the technical skills involved in setting up the Raspberry Pi with the Sense HAT I have included a section on how to go about this **(Raspberry Pi Learning Resources A) (Raspberry Pi Learning Resources C)**. This could be used by the teachers or in turn they are able to give this to the students for the students to carry out as an additional activity before they get started with the programming.

Whilst waiting on the parts for the robot to arrive I familiarised myself with the Raspberry Pi using the Sense HAT it was a great aid in learning the basics behind how the Raspberry Pi works. The Cardiff School of Computer Science and Informatics has already tried and tested a Raspberry Pi based outreach activity and the feedback was largely positive and undertaking this mini project enabled me to understand what is needed to create a successful outreach activity. Furthermore creating this worksheet gave me the opportunity to look a the national curriculum for computing.

The design of the worksheet takes into account the target audience, therefore, it uses colour coded code in order to make it easier for the student to understand the various aspects of the code. Even though there is potential to add more colour to the worksheet this could lead to increased printing costs. As the student proceeds through the worksheet they are challenged with a variety of exercises ranging from drawing images in pixels to reading information from the environment around them and through this process the student is learning some of the basics of computer programming. The worksheet then proceeds to give the students a range of challenges to further their practice and give them a chance to apply what they have learnt from the exercises.

Furthermore, there is the Arduino worksheet appendix 8.7. which involves the students learning how to use the Arduino to control certain electrical items such as LED's, Motors and Sensors. This activity was designed in order for the students to be able to interact with the components and actually be able to visualise moving parts when learning how to code. This worksheet also introduces the students to some basics of electronics, even though there is no subject that is taught in this specific field in forms one to three, this worksheet looks to introduce a new idea that may be involved in the national course curriculum at a future date. The design of this worksheet is progressive, introducing students to more complex ideas as the work through the exercises. The worksheet also contains some challenges that allow the students to think for themselves and apply what they have learnt in the previous exercises. This worksheet was designed around the components worked with during the project.

5. EVALUATION

In order to evaluate the quality and success of the outreach activity, I devised the two questionnaires (appendices 8.5. and 8.6.) one for the teachers and one for the students. These forms allow me to obtain feedback from all the parties involved in the engagement activity on their thoughts of how the activity was carried out, whether or not they gained from it and any future improvements that could be made to the activity for it to be more successful. Once this information was received and processed we would be able to assess the success of the activity as a whole which would allow us to further develop the activity to continue improving and changing it over time as simply running the same activity over and over again would not be enough to keep students engaged. With this questionnaire, I am able to find out whether there is potential in running an activity like this and if there is then I feel that there should be continuous development of the activity as this will keep the students hungry for more overtime. These questionnaires can be used with both the Sense HAT and Arduino worksheets.

Unfortunately, a visit to a school in which the activity could be performed and assessed was not possible, therefore, I was unable to obtain any feedback on the activity from the questionnaires. However, the worksheet (appendix B) was sent out to a teacher who's feedback was positive. To further evaluate the worksheets I looked at the computing course guidelines from key stages one to three and gave an explanation of how parts of the worksheet met the teaching outcomes outlined within the course specification guidelines. These details can be found in appendix D. After looking at the course specifications for each key stage I personally believe that even though this activity was initially set out for the students aged fourteen to fifteen (key stage three) it is more suited for a younger age group as it meets more of the learning outcomes for the lower key stages. I believe it is best suited for students of age eleven to twelve (key stage one). In order for the activity to be suitable for the intended audience, it would have to introduce more advanced concepts.

The students who take part in this activity will have an opportunity to see what a student who studies computer science, a course that they can apply for, is able to achieve. If the student is to find the activity interesting and is excited about the concepts introduced they will naturally want to learn more. Therefore, through this activity they are able to gain interest in studying at the Cardiff School of Computer Science and Informatics.

The initial goal of the project was to use the robot as a tool to teach, however, at this point in the project the robot does not have the functionality to act as an interactive tool that can be used in teaching. Due to this, the initial engagement activity will make use of the Sense HAT and Arduino worksheets created. Even though the project has not met its goal in time, a prototype on which future improvements can be developed has been created. In the creation of this prototype I have managed to make the motors spin in both directions and hence given the ability for the robot to manoeuvre in any direction. Furthermore, the sensors to use have been identified and any finding on how to connect them and use them has been made available. For example, how the PWM pins enable you to control the speed of the motors and how making the sensor read many readings and return an average in order to decrease the chance of errors. Furthermore, it is now known that when using a four wheel based chassis it is important that the chassis has been made properly as otherwise you will face problems with making the robot rotate and turn. Through the research undertaken in the project it has been made clear that there is no need for a robot that uses both the

Raspberry Pi and Arduino as this would only add an increased amount of difficulty in creating the system which is greater than the increased level of functionality it may give the robot. The code is extremely simple to understand therefore with little to no explanation someone with some knowledge of computer programming will be able to further develop this code. Meaning this code can easily be reused for any further projects of this kind.

The robots chassis is extremely durable and reliable, however, the connections can often come undone as the wiring is visible and left open. Moreover, none of the connections are soldered in place meaning they have more of a chance of becoming undone. Soldering elements would require health and safety training which was not feasible in the timeframe of the project. This leads to an increased amount of space being used on the chassis. Which makes it difficult to add any more functionality to the robot. However leaving the wiring open and unsoldered means that future modifications will be easier to make as this is just a prototype for future developments and if the activity does becomes obsolete the hardware can easily be repurposed for another cause.

The overall cost of the robot is outlined in the table below. Compared to the robots that are already on the market the cost is relatively low, however so is the level of functionality.

Part Description	Cost (£)	Source
Chassis (With Motors, Wheels and Battery Pack)	21.96	Ebay
Ultrasonic Distance Sensor (HC-SR04)	0.60	Amazon
Wires (x14)	1 (approx)	Amazon
Breadboard	1.44	Amazon
Batteries (4xAA)	1.77	Amazon

The total cost of producing one robot prototype is £26.77 which is approximately $40, which is considerably less expensive compared to most of the products currently on the market. Furthermore, if there was demand for more than one then these parts would need to be sourced at a wholesale price in order to further drive down the cost of the robot. Note, the cost involves one set of batteries that would come included with the robot, additional batteries would need to be purchased by the customers. The low cost of development is a positive as it allows for a larger budget for future developments still keeping with the aim of making a more affordable robot compared to what is currently on the market.

6. FUTURE WORK

As this project did not completely come to meet the goal it set out to achieve, future work would involve creating an interactive experience with the robot that could be used to teach the students about computer programming and electronics. In the short run, this would involve creating more worksheets similar to the ones already designed that introduce new concepts and allow students to engage with new pieces of hardware.

If I were to start this project again after gaining the knowledge and expertise I now have there would be a few things I would do differently. Firstly I would ensure that when choosing a chassis upon which to base the robot on a greater amount of research into the chassis would be done. Potentially it would be best if the chassis was designed from scratch as in that way any future modifications to the chassis would be easily made and the chassis could be made to be suited for the robot to be created. The next step would be to undertake health and safety training in order to use a soldering iron as this would allow for an increased amount of flexibility instead of being restricted by the size of the breadboard. With this increased amount of flexibility, the ability to add more sensors to the robot will exist. This would allow for the connection of an infrared sensor which could be used to develop the functionality for the robot to follow a line mapped out on the floor and use this to create an interactive board game that could be played by the students. Instead of connecting the HC-SR04 to a fixed position I would mount it to a servo motor so that it has the ability to rotate and read for obstacles to the left and right of the robot as well as directly in front. With this, the robot would be able to know whether to turn left or right dependent on the position of the obstacle instead of basing it on a random basis.

In terms of the motors I would continue to use direct current motors as they provide the functionality needed with a relative amount of simplicity, however, I would look to change the power source of the motors to a rechargeable power supply so that there is no need to keep replacing the batteries. Furthermore, this power supply would have to be capable of powering the motors at their full capacity instead of half which is what the current prototype does. Which would lead to a decreased upkeep cost making the product more attractive to schools. For controlling the motors using the Arduino I would use a motor shield such as the Arduino Motor Shield as this would allow for more of the input and output pins on the Arduino to be made averrable for connecting more sensors. Anyone who decides to take this project further will easily be able to gain the know how of what I have done so far if they follow the Arduino Worksheet created. I would also like to develop an object oriented approach to the code from the beginning so that new sensors and motor controllers can be added with greater ease.

Lastly instead of carrying out one outreach engagement activity per school, I would like the project to develop new material so that there would be continuous interaction between the schools and the university.

7. REFLECTION

The aim of this project was to create an outreach engagement activity to further inform students about the various aspects of computing. When starting the project the way in which I could approach it was open and the only restriction was to use products such as the Raspberry Pi or the Arduino as tools to aid the engagement activity. I made a decision to make a vehicle-based robot that the students could engage with, on reflection, I should have spent more time considering the approach, as at the time I did not posses the knowledge nor the experience to realise the difficulty of the project I had proposed. This phenomenon was made clear to me through each stage of the project as with every step forward I often had to take two steps back and reevaluate my approach to the problem. When proposing the project to my supervisor I was told clearly that what I had set out to do was ambitious as I had very little history with electronics in general and none whatsoever in the field of robotics.

Once the decision to use a vehicle-based robot was made, the next step was to decide on how to tackle the project and which methodology would work best for the project. As stated in the approach section, I initially thought that the best approach to use would be the waterfall method, however, this was not as appropriate as I initially thought and therefore I considered it appropriate to change it to using a more agile based approach but still maintaining some elements of the waterfall method. This process taught me that when undertaking a project you may think you know which way to tackle it is best but often the case may be that as the project continues on its timeline that you are able to realise that there are in fact better ways of approaching the project. Moreover, there may not be one specific approach that will be best suited to the project and you may have to combine aspects from various approaches in order to come up with a method that suits the project best.

Upon acquiring a Raspberry Pi and Arduino, I decided to learn the basics of each device. The previous, be it little, experience with electronics and physics was extremely useful at this stage as it gave me a small foundation to work on. This involved making basic circuits using both devices to control when an LED switched on and off, the brightness of the LED and further developed on my knowledge of electronics basics such as voltages, currents and resistance. Through this process I developed knowledge on the GPIO pins on each device and that involved knowing which pins can be used in various situations. For example, the PWM pins can be used to control the speed of the motors, therefore, it is important to connect the enable pins of each motor to a PWM compatible pin. This process taught me that it is important to start from the very basics and continue to develop a strong foundation on which you are able to continue developing the project on. With a better foundation, aspects of the project become easier to manage as you are able to for see problems and come up with better solutions to these problems. For example, initially I though that I would use a diode and a transistor to control the speed of the motor, but after developing this foundation it was clear to me that this would lead to a greater chance of problems as compared to using a PWM pin instead. This is a skill that I will look to apply to all of the work I do in the future as I believe that this small amount of work can go a long way to creating a deeper understanding and therefore being able to produce work of a better quality.

After conducting the research review, I came to the realisation that the plan proposed was ambitious because the products that existed on the market were highly more advanced than the product proposed. Even with this realisation I decided to continue with the project as proposed, this helped create an understanding that it is easy to feel like you are lost or down and out before even attempting to solve the problem, however, it is important to maintain a degree of persistence to overcome the challenge and reach the final goal. I have taken part in a vast number of mountain climbing activities in the past and those experiences have taught me that persistence is key to climbing to a summit and achieving your goal of conquering the mountain. In the case of this project that goal is to create an interactive vehicle-based robot that can be used as a tool in an outreach engagement activity to teach students the basics of computer programming.

The initial goal of the project was to create a tool using the Raspberry Pi, Arduino or any other similar device that could be used in an outreach engagement activity to teach students of ages 14-15 about computing. The initial idea was to create a robot that used both the Raspberry Pi and the Arduino. Although I must state that I overlooked the complexity and time that would be involved with creating such a system when I first set out to start the project. I was in-fact told by my supervisor that my plan was very ambitious but due to my high ambition and stubbornness to listen I still went ahead with my plan. This process taught me that it is always important to listen to what your mentors have to say as they have the experience and knowledge backing them that you may not have when starting out in a project. However, this does not go to say that being ambitious is a negative, as often it is this ambition of creating such a complex product that enables us to push ourselves as far as possible which results in a greater amount of innovation in the same time period. I personally believe that if I had not set my ambition so high, I would not have been able to achieve the same as I have with this project.

The design, implementation and testing stage involved a large learning curve. Most of the learning took part in the first iteration and due to the experience gained in this iteration the rest did not involve a large learning curve in comparison. It is important to note that the initial experience gained from creating a foundation upon which to work on was responsible for a shorter learning curve through each of the iterations. Furthermore I found that my passion for cars really helped me with this project and an example of that is where I tested the theory to see that four motors actually spun at a lower RPM compared to when only two were connected to the same power source. I got the idea for this test from my knowledge of rally cars, specifically the type of gearbox used in rally cars. This gearbox is called a dog-box and it allows the drivers to change gears without engaging the clutch. The gears in an everyday car have curved teeth that slide into place with other gears like a jigsaw puzzle hence you have to press the clutch to allow this teeth to align and smoothly come together. Whereas in a rally car the teeth are straight and sharp much like the ones in the DC motor. When these gears grind against each other they produce a sound which gains in frequency and pitch as the RPM increases. Therefore I recorded the sound of the motors spinning and used that information to determine that the four motors were in fact spinning at a lower RPM. I had undertaken a placement year, last year which unfortunately I was unable to complete, however this allowed me to go back home for a large period of time and in this time I was able to spend time with cars and gain the knowledge and experience that I now posses.

Throughout this process I have been able to apply my problem solving skills to the maximum as at every turn I was presented with another problem. This helped me to apply my problem solving skills and further improve on them. Moreover, it made me realise that I do in fact posses the patience and skill to tackle abstract problems that don't have a single fixed solution. One example of

this is when I had to edit the DCMotorBot library in order to add the functionality to control and power four motors. Another example of this is when I had to decide whether to use all four motors or just two. This was one of the biggest problems that I faced throughout the project. There was no one right way of doing it I had to be patient, experiment and apply my problem solving skills in order to come up with a solution.

Even though I had initially set out to make a four wheel based robot and ended up with a two wheel based robot, I would not think of it as a failure as in the process of trying to make the four wheeled robot work I was able to gain further understanding into topics that already interested me such as differential steering. I already knew what the underlying principle of differential steering was however with this project I was able to further my insight on this particular topic. The lesson learned from this experience is that even when things don't work and it may seem like you have not gained anything, you always do gain some insight and knowledge in areas that you may not have otherwise had. So even if you feel like you have wasted your time, nothing is really ever a waste of time and if you try and make things work you will gain some insight whether or not the final goal is achieved.

During the entire span of the project, I had to be extremely resourceful and often come up with DIY fixes for certain problems. For example, I wasn't sure on how I would attach the 9V battery to the chassis as it had to be attached in a manner so that it was easy to replace when it ran out of power. I experimented with double sided sticky tape, which did not work as the battery was too difficult to detach and new tape would be necessary every time the battery needed changing. Eventually, I used a rubber band that I found lying at home to hold the battery in place. Furthermore as mentioned earlier I did not have a full set of tools at my disposal hence instead of using a screwdriver I used a carrot peeler. This just goes to show that even if you don't have the specialist tools, if you are able to think outside the box, you may be able to come up with an alternative fix.

Looking back at the project as a whole I am really glad that I decided to try and do something that wasn't a traditional computer science project as it allowed me to explore a new field one in which I have now developed a great interest in and may look to continue further studies in. Furthermore, this project allowed me to apply knowledge from my passion of cars which made me very passionate about the entire project.

8. APPENDICES

8.1. Appendix A - External Material

The external material used in this project consist of:

- National Curriculum for Computing Key Stages 1 and 2

https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/239033/PRIMARY_national_curriculum_-_Computing.pdf

- National Curriculum for Computing Key Stages 3 and 4

https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/239067/SECONDARY_national_curriculum_-_Computing.pdf

8.2. Appendix B - Sense HAT Worksheet

TEACHERS SECTION

What you will need:
- Raspberry Pi with SD card and up to date version of Raspbian installed
- Keyboard
- Mouse
- Sense HAT
- Sense HAT Python library

What the students will do:
Through this worksheet students will gain an understanding of the Sense HAT hardware and its corresponding Python library. Students will learn how to use the LED matrix, collect data from sensors about their surrounding environment and combine these techniques in several mini projects.

What the students will learn:
- How to use IDLE (Python programming environment) to communicate with the Sense HAT.
- How to access and program the Sense HAT's inputs and outputs
- How to use the Sense HAT library to:
  - Display messages and images
  - Gather sensor data
  - Control orientation
  - Respond to movement
- How to use variables to store sensor data
- Use loops in order to repeat certain actions

What is the Sense HAT:
The Sense HAT is an add-on board for the Raspberry Pi and is attached to the Pi via the 40 GPIO pins. It has several in built sensors that have the ability to sense the following:
- Orientation (pitch, yaw and roll) via the accelerometer, 3D gyroscope and magnetometer
- Pressure
- Humidity
- Temperature

and then output this information on the built-in 8x8 LED matrix.

Note: The Sense HAT functions can all be found within the Fact Sheet provided.

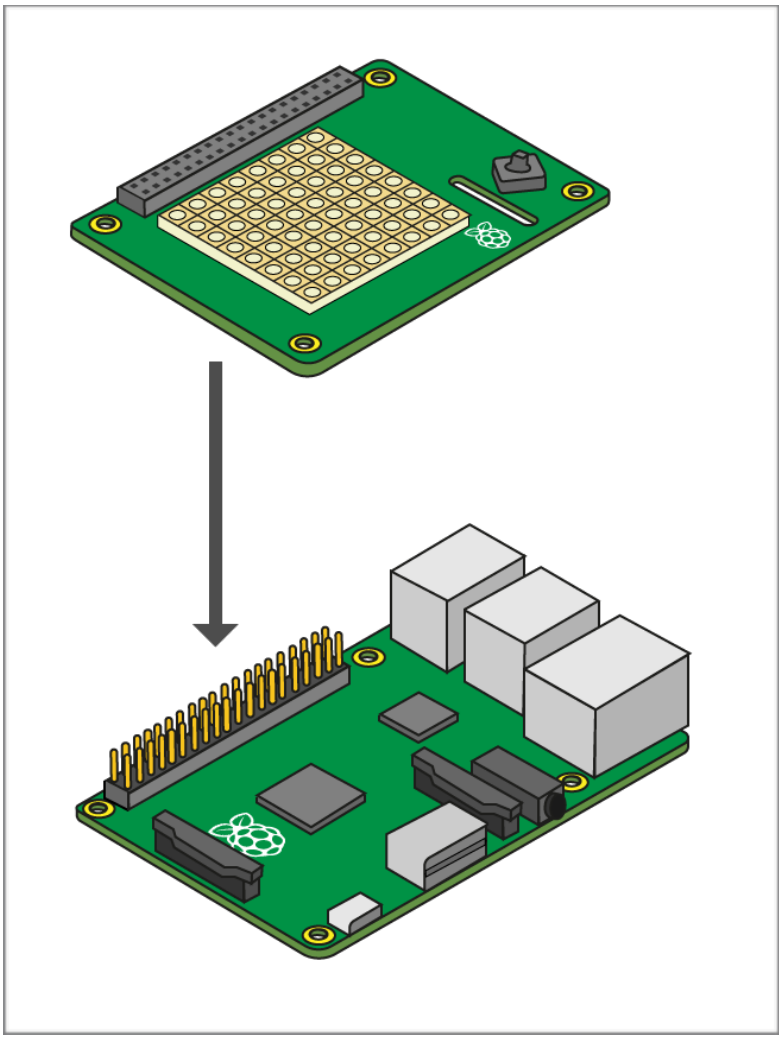
HOW TO SET UP RASPBERRY PI WITH SENSE HAT

Whats Required:
- SD Card
  - Pre-installed with noobs. If the card does not have it pre-installed you can visit the raspberry pi downloads page and download your free copy.
- Display and connectivity cables
  - A HDMI/DVI monitor or TV will work as a display for the Raspberry Pi.
- Keyboard and Mouse
- Power Supply
- Internet Connection
  - To update or download new software. Raspberry Pi can be connect to the internet via Ethernet or WiFi if you have a WiFi module.

Attaching the Sense HAT:
The sense hat comes with the following:

- 1 x GPIO pin extension header
- 4 x Hexagon stand-offs
- 8 x M2.5 screws

The following diagram shows how the Sense HAT fits the Raspberry Pi:



Plugging in the Raspberry Pi:

1. Slot the SD Card into the SD card slot in the Raspberry Pi.
2. Plug USB keyboard and mouse into the USB slots
3. Make sure your monitor/TV is switched on and that you have selected the correct input
4. Connect HDMI/DVI cable from Raspberry Pi to the TV/Monitor
5. Connect the ethernet cable to the ethernet port which is located near the USB ports on the Raspberry Pi.
6. Finally plug in the power, this will boot up your Raspberry Pi.
7. The follow the NOOBS installation guide.

Logging into the Raspberry Pi:

1. Once the boot process is complete the Pi will ask you to log in using a username and a password. The default log in username is pi and the password is raspberry.
2. After you have correctly entered the log in details, you will see the command line prompt, pi@raspberrypi-$
3. To start the graphical user interface simply type in the startx command

Software Setup:

1. Update and upgrade the software on the Raspberry Pi by entering the following commands into a new terminal window:

```
sudo apt-get update
sudo apt-get upgrade
```

2. Install the Sense HAT software package:

```
sudo apt-get install sense-hat
```

3. Lastly reboot the Raspberry Pi for the changes to take effect:

```
sudo reboot
```

4. In order to open the IDLE programming environment you can either type the following command into a terminal window or you can navigate to the start menu and the the programming sub menu and find and open python 3.

```
sudo idle3
```

EXERCISE 1 - Displaying Text

In this exercise you will write a program that scrolls get across the LED matrix. This program includes two critical lines which import the Sense HAT software and create an object sense which represents the Sense HAT.

```
from sense_hat import SenseHat

sense = SenseHat()
```

The next line is the one that actually makes the Sense HAT do something.

```
sense.show_message("Hello my name is…")
```

You are able to change the message to whatever text you wish it to display. Try get it to display "Hello my name is …. and I am … years old " filling in the gaps with your name and age.

The sense.show_message command can include some extra parameters which will alter the behaviour of the message.

Parameter	Effect
scroll_speed	How quickly the text moves on the LED matrix. The larger the number the slower the speed
text_colour	Alters the colour of the text and is defined by using 3 values for Red, Green and Blue. Each value can be between 0 and 255.
back_colour	Alters the colour of the background. It is defined by the same three values as with text_colour.

The program below will show a single red "A" on a blue background at a speed of 0.2 which is the default.

```
from sense_hat import SenseHat

sense = SenseHat()

sense.show_letter("A", text_colour=[255, 0, 0], back_colour=[0, 0, 255], scroll_speed=0.2)
```

Now we will try and display one letter at a time with a delay between displaying each letter. Therefore we will need to use a new function the sleep function which exists in the time library.

```
from sense_hat import SenseHat
import time
sense = SenseHat()

sense.show_letter("A", text_colour=[255, 0, 0])
time.sleep(1)
sense.show_letter("B", text_colour=[255, 0, 0])
time.sleep(1)
sense.show_letter("C", text_colour=[255, 0, 0])
time.sleep(1)
sense.clear()
```

EXERCISE 2 - SENSING THE ENVIRONMENT

In this exercise we will learn how to use the Sense HAT to sense the conditions around us.There are three sensors built-in to the Sense HAT that can measure the following conditions in the surrounding environment:

- Temperature Sensor
- Barometric Pressure Sensor
- Humidity Sensor

To read the data from these sensors we use the following three commands:

```
sense.get_temperature() - This will return temperature in Celsius
```

```
sense.get_pressure() - This will return pressure in millibars
```

```
sense.get_humidity() - This will return the humidity as a percentage
```

Now, lets create a program that would keep text scrolling across the LED Matrix to continuously keep people informed about the current conditions. It would be helpful to have some visual indication for example when the temperature goes above a certain value.

```
from sense_hat import SenseHat
sense = SenseHat()

while True:
        temperature = sense.get_temperature()
        pressure = sense.get_pressure()
        humidity = sense.get_humidity()

        temperature = round(temperature, 1)
        pressure = round(pressure, 1)
        humidity = round(humidity, 1)

        if temperature > 18.0 and temperature < 26.0:
                bg = [0, 100, 0]
        else:
                bg = [100, 0, 0]
        scrollmsg = "Temperature = %s, Pressure = %s, Humidity = %s" (temperature,pressure, humidity)

        sense.show_message(scrollmsg)
```

Click File —Save As, and give the program a name like environment.py, then press F5 to run the program. You should see the temperature, humidity and pressure scroll across the LED Matrix.

Can you get the message to scroll at a different speed? The scroll speed argument can be used to change the speed at which the text scrolls across the LED display.

Here you can see the use of a while loop, this means the code in this loop keeps repeating itself as long as the condition is met. Therefore setting the condition to True means that this code will always keep running. Inside the loop the first three lines of code obtain the values from the respective sensors. The next three lines use the round function to round these values off. The next three four lines of code tell the Sense HAT to change the background to red when the temperature is out of the range of 18 to 26 degrees and otherwise keep it green.

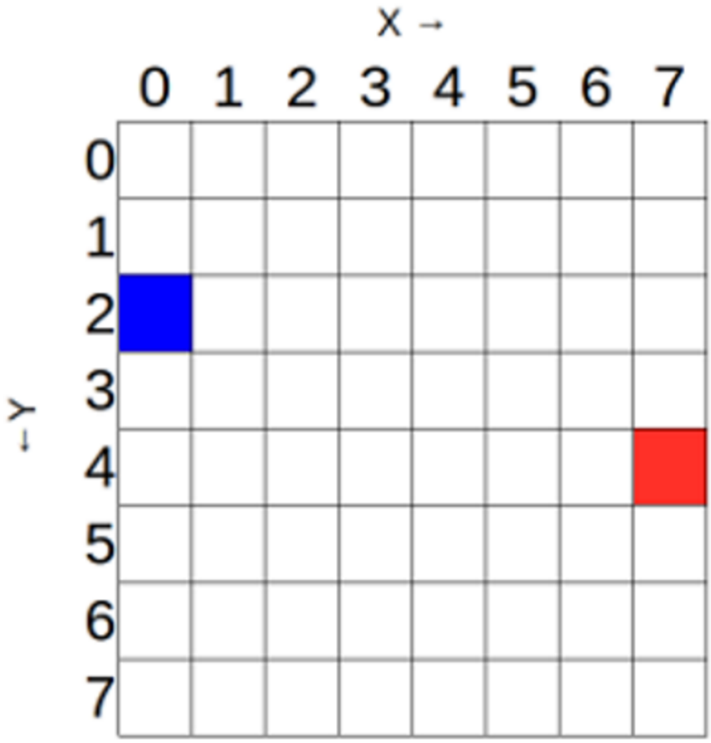
EXERCISE 3 - DISPLAYING IMAGES ON THE LED MATRIX DSIPLAY

In this exercise we will learn how the LED display can display images and not just text. The LED matrix is made up of many LED's therefore we can control each one to create an image.

One way to do it is to use the following method to set each LED individually:

```
sense.set_pixel()
```

Before we get ahead of ourselves lets understand how each pixel is described. The Sense Hat uses a co-ordinate system like the one pictured below; It may be similar to what you might have used in geography. However there are a couple of differences, most impertinently the numbering does not start with 1 but with 0. Also, the origin is located in the top left corner of the matrix and not the bottom left like you may be used to.



The blue pixel has the coordinates (0,2). So can you tell what the coordinates are for the red pixel?

In order to display the above image on our LED matrix display we would need to do the following:

```
from sense_hat import SenseHat

sense = SenseHat()

sense.set_pixel(0, 2, [0, 0, 255])
sense.set_pixel(7, 4, [255, 0, 0])
```

As you can see from above, the sense.set_pixel() method takes three inputs. The first is the x-coordinate of the pixel, the second is the y-coordinate and the last three figures in square brackets are to set the colour of the pixel.

Now this method works perfectly fine but it could get rather complicated when you want to set more than just a few pixels (LED's). With the sense.set_pixels() method we can set more than just one pixel at a time. Notice you may think that I have repeated the same method as the discussed earlier but this one is sense.set_pixel**s**. With this method we simply give a list of colour values for each of the pixels in the matrix. It would look something like the following:

```
sense.set_pixels([[255, 0, 0], [0, 255, 0], [0, 0, 255], [255, 0, 0],......])
```

but even this would be extremely complicated and take too much time. Instead we assign various colours to different variables.

```
r  = [255, 0, 0]
o = [255, 127, 0]
y = [255, 255, 0]
g = [0, 255, 0]
b = [0, 0, 255]
i  = [75, 0, 30]
v = [159, 0, 255]
e = [0, 0, 0]
```

Do you know what colours all the variables represent, you may have a hard time with the e variable. This variable is used for an empty space and represents the colour black.

Now you might ask what is the use in creating this colour palette? We can now use this palette to to describe out matrix as shown below:

```
image = [
e,e,e,e,e,e,e,e,
e,e,e,r,r,e,e,e,
e,r,r,o,o,r,r,e,
r,o,o,y,y,o,o,r,
o,y,y,g,g,y,y,o,
y,g,g,b,b,g,g,y,
b,b,b,i,i,b,b,b,
b,i,i,v,v,i,i,b
]
```

The next step is to give this image list to the sense.set_pixels() method so that the image can be drawn on the LED matrix display. So can you take all these bits of code and put them into a program in order to get this image to show up on the LED matrix display?

CHALLENGES

1. Instead of getting the Sense HAT to display the letters A, B, and C one after another can you try and get your name to show up in the LED matrix display.
2. Try and make your name appear in blue on a blue background.
3. In exercise 2 we round off the values of temperature, humidity and pressure but if for example we needed more exact figures and wanted to know the values rounded of to the first 3 significant figures, how would this be possible? Try and implement a program that outputs these three values to 3 significant figures.

8.3. Appendix C - Sense HAT Fact Sheet

FACT SHEET

Program vs Algorithm: An algorithm is just a guideline to solve any particular problem. Whereas a computer program is the actual code for solving the problem which is written in a language that the computer can understand. (PYTHON!) *Give example of making cake. Found below under the relevant examples heading.*

Variable: A variable is a reserved memory location that is used to store values. Therefore when you create a variable you simply reserve some space in memory. In python variables do not need to be explicitly declared as this declaration takes place automatically when a value is assigned to a variable.

Function: A function is an organised block of code that can be used to perform a single action.

Command: A command is simply an instruction given by a user to a computer in order to do something, such as run a program.

While Loops:
The structure of a while statement is:

```
while <condition>:
        <do stuff>
```

The program asks the question is the given condition met or not? (i.e. is it true or false?). If the condition is met then it executes the code in the <do stuff> section. Once it completes executing the code within the loop it goes back to the start and checks if the conditions is still true and if it is it will repeat the process until the condition is not met.

IF Statement:
The structure of an if statement is:

```
if <variable> == <condition>:
        <do stuff>
```

The program checks to see if a certain variable meets the condition set and if these match or its true then the program executes the code within the <do stuff> section. If however the condition is not met then the code is not executed.

<u>RELEVANT EXAMPLES</u>

Baking a Cake Example

Baking a cake algorithm:
1. Get ingredients
2. Mix ingredients
3. Put in baking tray
4. Bake in oven
5. Let cake cool
6. Serve or refrigerate

Baking a cake program (recipe):

- You will need the following:
  - 2 cups flour
  - 2 cups sugar
  - 1 egg
  - etc…
- Then do the following:
  - Sift flour and sugar in mixing bowl
  - Put in liquid ingredients and mix throughly
  - Pour into a 12” cake pan
  - Bake at 280 degrees for 30-35 minutes
  - Remove cake from pan and leave to cool
  - etc…

The distinction you are trying to make here is that the algorithm is extremely general, but it does give the idea of what is involved in the process, and what steps are required and in which order. Where as the program (recipe) is very detailed for making a certain kind of cake and it will actually produce it unlike the algorithm. Therefore it can be seen as each cake program (recipe) will use the general structure of the cake algorithm but each recipe will vary and produce different outputs.

Add Sense HAT Functionality:

```
from sense_hat import SenseHat
sense = SenseHat()
```

LED Matrix Functions:

sense.set_pixel(0,0,255,0,0)	Sets the top left LED to the colour red
sense.show_letter(“J”,0,0,255)	Displays the letter “J” on the screen in blue
sense.show_message(“msg”,text_colour=[0, 255, 0])	Displays the message “msg” on the LED matrix in green
sense.load_image(“creeper.png”, redraw=True)	Loads an 8x8 image “creeper.png” and displays it on the LED matrix
sense.clear()	Clears the LED matrix by switching off all the LED’s
sense.set_rotation(r=0)	Sets the rotation of the LED matrix
sense.set_pixels(pixelList)	Uses pixellist to draw a picture, each item is an [R,G,B] list

Movement Functions:

yaw,pitch,roll = sense.get_orientation().values()	Gets orientation data and stores them as yaw, pitch and roll
m_x, m_y, m_z = sense.get_compass_raw().values()	Gets compass data and stores them as m_z, m_y, m_z
x, y, z = sense.get_accelerometer_raw().values()	Gets the accelerometer data and stores them as x, y, z
g_x, g_y, g_z = sense.get_gyroscope_raw().values()	Gets the orientation data and stores them as g_x, g_y, g_z

Temperature, Humidity and Pressure Functions:

t = sense.get_temperature_from_humidity()	Uses the humidity sensor to get temperature and stores it as t
t = sense.get_temperature_from_pressure()	Uses the pressure sensor to get temperature and stores it as t
h = sense.get_humidity()	Measures the humidity and stores it as h
p = sense.get_pressure()	Measures the pressure and stores it as p

Full Code for Exercise 3:

```
from sense_hat import SenseHat

sense = SenseHat()

r  = [255, 0, 0]
o = [255, 127, 0]
y = [255, 255, 0]
g = [0, 255, 0]
b = [0, 0, 255]
i  = [75, 0, 30]
v = [159, 0, 255]
e = [0, 0, 0]

image = [
e,e,e,e,e,e,e,e,
e,e,e,r,r,e,e,e,
e,r,r,o,o,r,r,e,
r,o,o,y,y,o,o,r,
o,y,y,g,g,y,y,o,
y,g,g,b,b,g,g,y,
b,b,b,i,i,b,b,b,
b,i,i,v,v,i,i,b
]

sense.set_pixels(image)
```

8.4. Appendix D - Matching National Curriculum Criteria

HOW THE SENSE HAT WORKSHEET LINKS TO THE COMPUTER SCIENCE CURRICULUM

This document only states the outcomes that were met my the worksheet in Appendix B. Please refer to the course guidelines in Appendix A for further information.

Key Stage 1:

Understand what algorithms are; how they are implemented as programs on digital devices; and that programs execute by following precise and unambiguous instructions.
Through out the worksheet the students are required to create algorithms. In exercise 1 the students write an algorithm that allows them to output and display a letter to the LED matrix on the Sense HAT. This would be a good place to explain what an algorithm is and how it differs from a computer program *(Explanation contained within fact sheet)*.

Create and debug simple programs.
In all three exercises the students are required to write and create multiple simple programs. There are no direct debugging exercises available but the students will learn how to debug as they are likely to make mistakes as they go along. These mistakes will be more syntax errors than anything else which should be easy to fix as mostly they will involve simple spelling mistakes or the omission of a character such as a colon.

Use technology purposefully to create, organise, store, manipulate and retrieve digital content.
Exercise 2 is a perfect example of where this objective is met as the students are required to get digital data from the sensors, store the data within a variable and then retrieve the data and display it in a useful manner. They also learn about the organisation of data through the use of the IF function *(If the temperature is within a certain range then do this else do something else)*.

Recognise common uses of information technology beyond school.
The students will learn about how the various sensors are used and where such sensors exist in the real world and what their applications are. It is important to create discussion as the students work on the worksheet asking them where such a program would be useful with a real world example.

Key Stage 2:

Design, write and debug programs that accomplish specific goals, including controlling to simulating physical systems; solve problems by decomposing them into smaller parts.
In all three exercises the students are required to write and create multiple programs that all accomplish specific end goals. That involve controlling some physical system such as the LED matrix and the various sensors that are present. There are no direct debugging exercises available but the students will learn how to debug as they are likely to make mistakes as they go along. These mistakes will be more syntax errors than anything else which should be easy to fix as mostly they will involve simple spelling mistakes or the omission of a character such as a colon. In exercise 3 the students are required to take lots of small bits of code and put them together in order to create a program that displays an image on the LED matrix display.

Use sequence, selection and repetition in programs; work with variables and various forms of input and output.
The use of selection and repetition both exist in exercise 2 as the students are required to use a while function (repetition) and the if statement (selection). Throughout the worksheet they assign

data to various variables and then output this data. In exercise 3 the students work with variables assigning colours to various variables. Then using these variables effectively.

Use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs.
This aspect of learning can be taught through discussion of the worksheet and to see if the students can explain bits of the code that lead to certain desired actions being carried out *(e.g. the if statement in exercise 2)*. In order to teach the students how to detect errors the teacher must purposefully make certain errors when demonstrating the code and get the students to point out why the code will not work. In exercise 3 the students are challenged to notice the difference between the two methods sense.set_pixel and sense.set_pixels this helps them think about debugging and how often small errors are often the cause for a program not to compile and run.

Select, use and combine a variety of software on a range of digital devices to design and create a range of programs, systems and content that accomplish given goals, including collecting, analysing, evaluating and presenting data and information.
The students will use various software including the standard python library, the Sense HAT library, linux and the python IDE on the Raspberry Pi and the Sense HAT to create a range of programs that accomplish given goals. This will involve them collecting data (from sensors), analysing and evaluation it (if function) and then presenting it.

<u>Key Stage 3:</u>

Use two or more programming languages, at least one of which is textual, to solve a variety of computational problems; make appropriate use of data structures [for example, lists, tuples, tables or arrays]; design and develop modular programs that use procedures or functions.
Throughout the worksheet the students will use the Python programming language which is in fact a textual programming language. They will use it to solve a variety of computational problems as they work through the three exercises.

Understand how instructions are stored and executed within a computer system; understand how data of various types (including text,sounds and pictures) can be represented and manipulated digitally, in the form of binary digits.
In exercise 1 students are expected to learn how to display text on the LED matrix display. Furthermore in exercise 3 the students are required to display images on the LED matrix display They are able to experiment by manipulating the colours or the colour of each individual pixel.

Undertake creative projects that involve selecting, using and combining multiple applications, preferably across a range of devices, to achieve challenging goals, including collecting and analysing data and meeting the needs of know users.
This worksheet should get the students feeling creative as they will be able to see the results in real time which will help them think about creative projects that they may want to undertake. They will understand the concept of using multiple devices to accomplish a certain end goal. Furthermore they will collect digital data from sensors and then analyse and organise the data and finally present it in a manner that is useful to the end user.

8.5. Appendix E - Teachers Feedback Form

Teachers Feedback Form

The following questionnaire is designed to help assess the value of the Outreach Engagement activity. The results will help determine the future of the activity. Please fill out the form as completely as you can and where there are multiple options please fill in your choice as clearly as possible. Thank you for your help.

School: **Department:**

1. What date did the engagement activity take place?

2. Did you find the activity useful?

   ☑ Very Useful
   ☑ Fairly Useful
   ☑ Neutral
   ☑ Fairly Unuseful
   ☑ Very Unuseful

3. Please use the space below to describe and particular positive or negative aspects of the activity.

4. What skills if any do you believe this project helped your students to develop?

4. Please us the space below to suggest any improvements that can be made to the activity?

5. Does this activity follow the general guidelines for the course?
   ☑ Yes
   ☑ No

6. If no, what improvements would you suggest be made to the activity?

7. Would you be happy to be interviewed about the activity, if yes, then please include your contact details.

8.6. Appendix F - Students Feedback Form

Students Feedback Form

The following questionnaire is designed to help assess the value of the Outreach Engagement activity. The results will help determine the future of the activity. Please fill out the form as completely as you can and where there are multiple options please fill in your choice as clearly as possible. Hope you enjoyed the activity and thank you for your help.

School: **Key stage:**

1. What date did you take part in the engagement activity?

2. Did you find the activity useful?

   ☑ Very Useful
   ☑ Fairly Useful
   ☑ Neutral
   ☑ Fairly Unuseful
   ☑ Very Unuseful

3. On a scale of 1 – 5 which (1 being negative and 5 being positive) what would you rate this activity?

   ☑ 1 ☑ 2 ☑ 3 ☑ 4 ☑ 5

4. What skills did this activity teach you and which skills were you able to improve on?

5. On a scale of 1 – 5 (1 being negative and 5 being positive) how much did you know about computing before this activity?

   ☑ 1 ☑ 2 ☑ 3 ☑ 4 ☑ 5

6. On a scale of 1 – 5 (1 being negative and 5 being positive) how much do you know about computing after the activity?

   ☑ 1 ☑ 2 ☑ 3 ☑ 4 ☑ 5

7. Would you change anything about how the activity is run?

8.7. Appendix G - Arduino Worksheet

Arduino Worksheet

Exercise 1

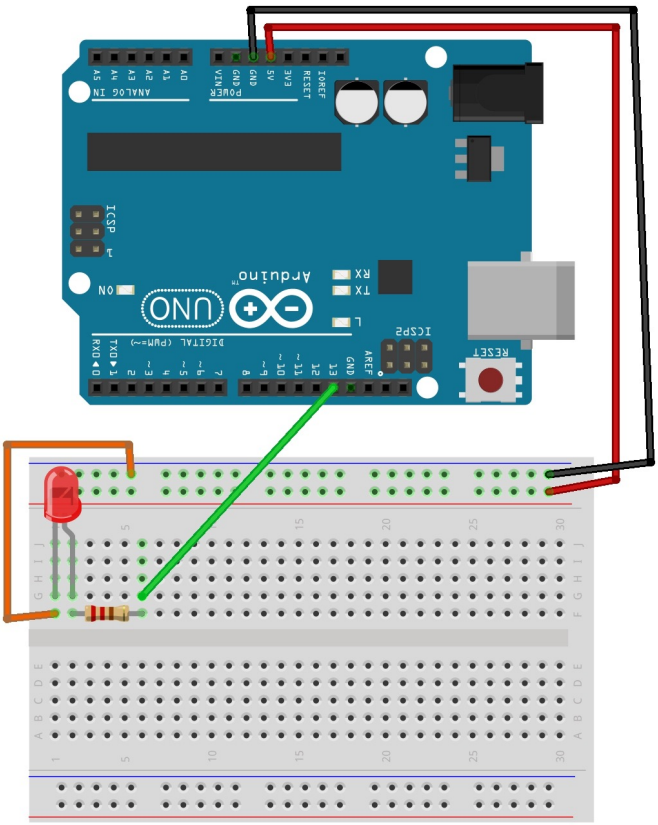
This exercise will teach you how to connect and control an LED using an Arduino. You will create a program to make the LED blink on and off.

For this exercise you will need the following:

- An Arduino UNO
- One LED
- One 220 Ohm Resistor
- One Breadboard
- Solderless Breadboard Jumper Cables

Building the Circuit

Use the schematic below as reference on how to set up the circuit.



Make sure that the Arduino is not connected to any power source when making this circuit. The first step is to connect the 5 volt output from the Arduino to the positive rail on the breadboard. The next thing to do is to connect the negative rail of the breadboard to the ground pin on the Arduino. Next place the LED into the breadboard with the positive leg (longer leg) called the anode on the right hand side and the negative leg (short leg) called the cathode on the left hand side. Next connect any side of the resistor to the positive leg as shown above. The final step is to make the connections to

the LED. This involves connecting a wire from the other side of the resistor the the positive rail and from the cathode of the LED to the negative ail on the breadboard.

Writing the Code

Once you have created the circuit above you can plug your Arduino into the computer and start the Arduino IDE software. Then write out the code below into a sketch.

```
void setup() {
        pinMode(13, OUTPUT);
}

void loop(){
        digitalWrite(13, HIGH);
        delay(1000);
        digitalWrite(13, LOW);
        delay(1000);
}
```

In the above code we have two functions the setup() section and the loop() functions. The setup() function is called when the script starts. It is used to initialise pin modes, variables and objects that use various libraries. This setup() function will only run once each time the Arduino is powered up or each time it is reset. The loop() function as you can guess from it's name, keeps being executed consecutively.

The first line of code in the setup() function is used to initialise pin 13 as an output pin:

```
pinMode(13, OUTPUT);
```

Within the loop() function the LED is first switched on using the following line of code:

```
digitalWrite(13, HIGH);
```

This line of code means that pin 13 is supplied with a high voltage (5 volts). Then the Arduino is instructed to wait for 1000 milliseconds (1 second) using the line of code below, therefore leaving the LED switched on for 1 second.

```
delay(1000);
```

The next step is to switch the LED off and this is done using the following line of code:

```
digitalWrite(13, LOW);
```

This takes the voltage of pin 13 back to a a low value (0 volts). The section of code in the loop() function keeps looping over making the LED blink on for a second and then off for a second.

Now that you have learnt how to make an LED blink using the Arduino, your challenge is to make the LED switch on for 5 seconds and only go off for two seconds.

Exercise 2

This exercise will teach you how to connect and control DC motors using an Arduino. The exercise will go through the basics of connecting the motors to the Arduino via a h-bridge. You will then go on to learn how to change the speed of the motors.

For this exercise you will need the following:

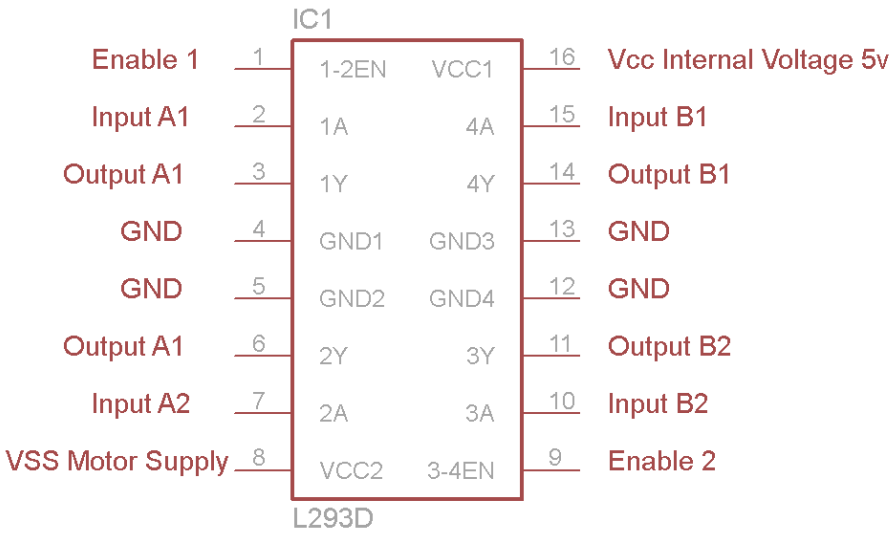
- An Arduino UNO
- One DC Motor
- One L293D H-bridge
- One Breadboard
- Solderless Breadboard Jumper Cables

A DC (Direct Current) motor is one of the most commonly used motors. They have two terminals, one positive and the other negative. If you connect a battery pack to these two terminals the motor will start to spin in one direction and if you switch the connections around the motor will spin in the other direction.

In order to control the direction the motors spins without having to physically change the connections around we can use a H-Bridge circuit.

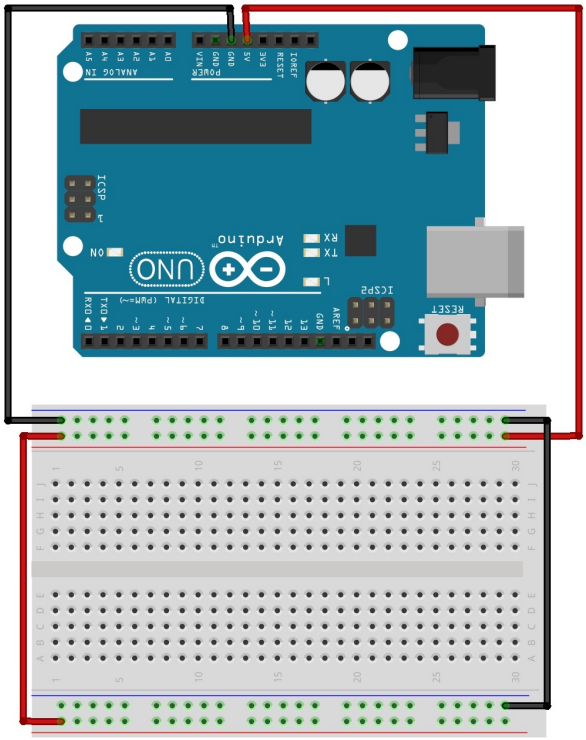
Building the Circuit

The diagram below shows the pins that exist on the H-Bridge.

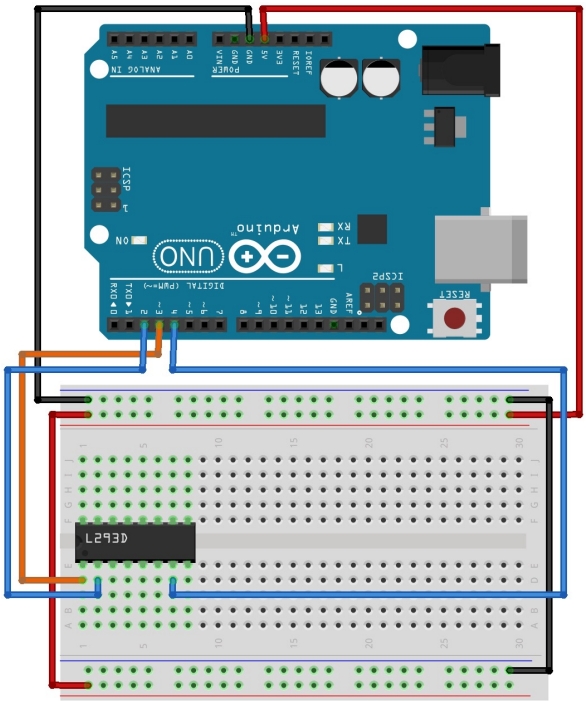


As you can see on each side there are two input pins, two output pins and one enable pin. This is because the L293D is capable of controlling two motors, one from the right hand side and the other from the left hand side. However for the purpose of this exercise we will only use one side of the L293D as we are only going to control one motor.

Lets go through all the connections step by step. The first connections to make are from the Arduino to the breadboard. The 5V output and ground from the Arduino must be connected to the positive and negative rails respectively as shown below. We must also make sure both sides of the breadboard are connected to the Arduino.

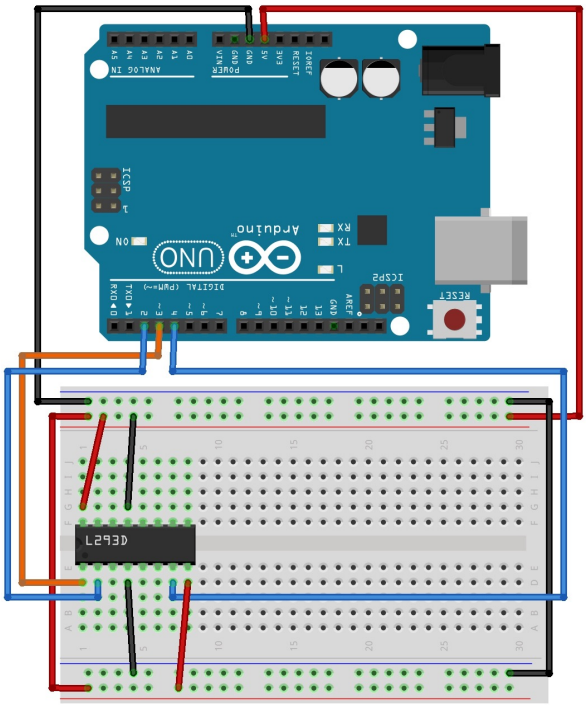


Once this is complete we must connect the L293D to the breadboard and to the Arduino.

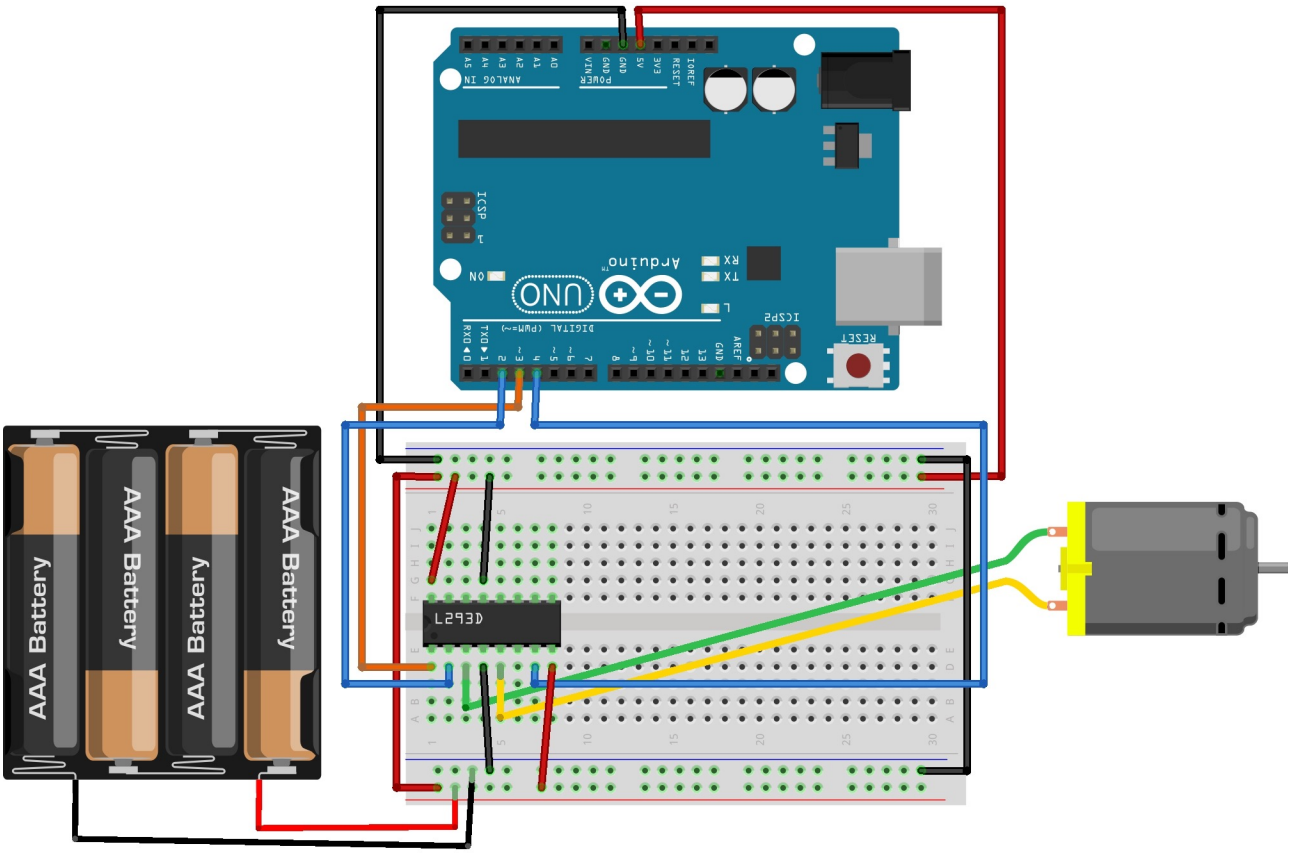


As you can see the the Arduino is connected to the L293D by three wires, this is because there are three input pins for each motor. For the purpose of this tutorial we will only use one side of the L293D as we are only going to control one motor. We will connect Input A1 to pin number 2 and Input A2 to pin number 4 and lastly the Enable 1 to pin number 3.

Pins 2 and 4 are digital pins which can take wither a value of LOW or HIGH (on or off), whereas pin 3 needs a pulse-width modulated (PWM) signal to control the speed of the motor. Therefore in order to set the values of pins number 2 and 4 we will use the digitalWrite() function used in the exercise above but to set the value of pin number 3, we will use the analogWrite() function. Each side of the L293D must be grounded and connected to the positive rails for a power source. Even though we are not using one side of the L293D it is good practice to do this.



Now lets introduce the DC motor to the the circuit above. The only connections left to make are the output connections to the motor and to connect the battery pack in order to power the motor.



Writing the Code

Once you have created the circuit above you can plug your Arduino into the computer and start the Arduino IDE software. The first step is to declare the variables relating to the pin numbers. This is done before the setup() function in the code.

```
int inputA1 = 2;
int inputA2 = 4;
int enable1 = 3;
```

Next, we must initialise these pins in the setup() function.

```
void setup() {
        pinMode(inputA1, OUTPUT);
        pinMode(inputA2, OUTPUT);
        pinMode(enable1, OUTPUT);
        analogWrite(enable1, 255);
}
```

After we have setup all the variables and initialised the pins we must create some code to control the state of the input pins in order to drive the motor in either direction.

```
void loop() {
        digitalWrite(inputA1, HIGH);
        digitalWrite(inputA2, LOW);
        delay(3000);
        digitalWrite(inputA1, LOW);
        digitalWrite(inputA2, HIGH);
}
```

This section of the code is split into two parts by the call of the delay function. The first two lines of code will make the motor spin in one direction and after a three second delay the motor will spin in the other direction. Now you can upload this code onto your Arduino and watch the motors spin in either direction!

Challenge

Now that we have learnt how to control the direction the motor spins in, we only need to figure out how to control the speed the motor spins at. Have a look through the code above and try and find out which bit of code can be changed in order to achieve this effect.

Exercise 3

This exercise will teach you how to connect and use the HC-SR04 ultrasonic distance sensor to the Arduino and how to read values from it on the serial monitor. Furthermore it will teach you how to use these values to control when to switch on and off an LED.

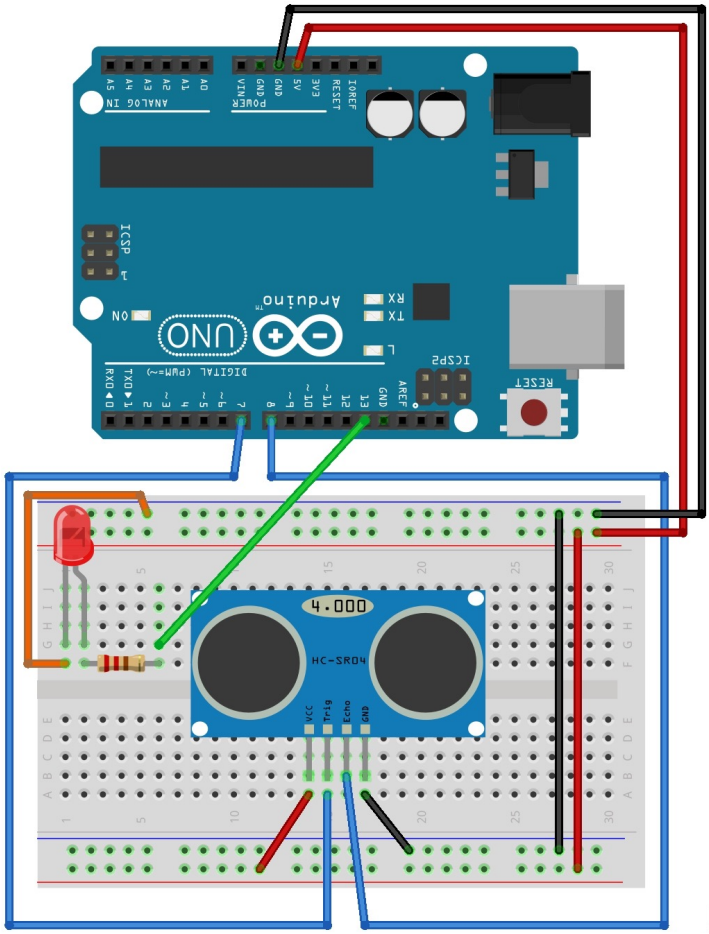
For this exercise you will need the following:

- An Arduino UNO
- One HC-SR04 Ultrasonic Sensor
- One LED
- One 220 Ohm Resistor
- One Breadboard
- Solderless Breadboard Jumper Cables
- The NewPing Library (download at: http://playground.arduino.cc/Code/NewPing#Download)

The HC-SR04 works using sound waves, it emits a sound wave and waits for the reflection of the sound wave to bounce of any surfaces, this reflection of the sound is then captured by the sensor and used to calculate how far away the object that reflected the sound wave is. You may have seen this concept used in submarines when they use sonar. Can you think of an animal that uses this concept? (Bats).

Building the Circuit

The first step here is to build the same circuit that you built in exercise 1 using the LED and the resistor. Once you have completed that place the HC-SR04 on the other side of the breadboard as shown below.



The HC-SR04 has four pins, there is a pin that receives the 5 volts of power the sensor requires to operate there is another pin that grounds the sensor, known as the ground pin. The other two pins are used to emit the sound wave and capture the reflected sound wave. The pin that emits the sound wave is know as the trigger pin and the one that captures it is known as the echo pin. So the power and ground pins connect to the positive and negative rails respectively and the echo and trigger pins connect to any of the pins on the Arduino in this case we will be using pins 7 and 8.

Writing the Code

In order to use the HC-SR04 you will need to use the NewPing library you downloaded earlier. Therefore we have to tell the Arduino that we are going to use this library. In order to do that we use the line of code below:

```
#include <NewPing.h>
```

Now that the Arduino knows that we are using this library , we need to set up an object for this library and some variables for the pins the HC-SR04 is connected to. To do this use the following code:

```
int trigPin = 7;
int echoPin = 8;
int redLed =13;
int maxDistance = 200;
int safeZone = 5;
int distance = 0;
NewPing ultraSonic(trigPin, echoPin, maxDistance);
```

The first three lines declare variables for all the pins that are being used on the Arduino. Pin seven is being used by the trigger pin on the HC-SR04, pin eight is connected to the echo pin and the led is connected to pin thirteen. The maxDistance variable is used to tell the HC-SR04 up until what distance it should read for objects. The safeZone variable uses will be clear later in the exercise. The last line of code declares an object of the NewPing library that is called ultraSonic using these variables as parameters. The next step is to create our setup function and our loop function:

```
void setup()
{
        pinMode(redLED, OUTPUT)
}

void loop()
{
        distance = ultraSonic.ping_cm();
        if (distance >=  safeZone || distance == 0){
                digitalWrite(redLED, LOW)
        }
        else{
                digitalWrite(redLED, HIGH)
        }
}
```

All that needs to be done in the setup function is to declare that pin thirteen is being used as an output pin the rest is taken care for us when we initialised the ultraSonic object of the NewPing library. In the loop function of the code we first measure the distance away an object is, if there is one. We then check to see if this obstacle is out of the safe zone we declared earlier under the safeZone variable name and we make sure that if the distance read is zero then the LED should remain switched off. If anything but this is to occur, i.e. an obstacle that is within the safe zone then the LED should switch on.

Now plug in your Arduino and upload the code and move your hand in front of the sensor to get the LED to switch on.

Challenge

Try to increase how close an object needs to be in order for the LED to switch on. Instead of just getting the LED to switch on can you get it to blink using what you learnt in exercise one.

9. REFERENCES

Biggs, J. 2014. Up Close With Play-i's Bo, The Lovable Xylophone-Playing Robot [Online]
Available at: http://techcrunch.com/2014/01/18/up-close-with-play-is-bo-the-lovable-xylophone-playing-robot/
[Accessed: 29th March 2016].

Bourque, B. 2015. Arduino vs Raspberry Pi: Mortal Enemies or Best Friends [Online]
Available at: http://www.digitaltrends.com/computing/arduino-vs-raspberry-pi/
[Accessed: 30th Feburary 2016].

Booth, H. 2014. Cubelets [Online].
Available at: https://robottestkitchen.com/category/cubelets/page/3/
[Accessed: 3rd April]

Burns, C. 2014. LEGO MINDSTORMS EV3 Review [Online].
Available at: http://www.slashgear.com/lego-mindstorms-ev3-review-03327308/
[Accessed: 2nd May 2016].

Choi, J. 2014. The complete guide to buying a robot for your kids [Online]
Available at: http://qz.com/309365/the-complete-guide-to-buying-a-robot-for-your-kids/
[Accessed: 3rd April]

Circuit Basics A. 2016. Basics of the SPI Communication Protocol [Online].
Available at: http://www.circuitbasics.com/basics-of-the-spi-communication-protocol/
[Accessed: 1st March 2016].

Circuit Basics B. 2016. Basics of the I2C Communication Protocol [Online].
Available at: http://www.circuitbasics.com/basics-of-the-i2c-communication-protocol/
[Accessed: 1st March 2016].

Codie Facebook, Codie 2016. About [Online].
Available at: https://www.facebook.com/codietherobot/info/?tab=page_info
[Accessed: 29th February 2016].

Eckel, T. NewPing Library for Arduino [Online].
Available at: http://playground.arduino.cc/Code/NewPing
[Accessed: 28th March 2016]

Geer, D. 2014. Science and Math in Engaging Ways [Online].
Available at: http://www.scientificamerican.com/article/4-robots-that-teach-children-science-and-math-in-engaging-ways/
[Accessed: 28th February 2016]

Hughey, D. 2009. Comparing Traditional Systems Analysis and Design with Agile Methodologies [Online]. Available at: http://www.umsl.edu/~hugheyd/is6840/waterfall.html
[Accessed: 26th February 2016].

Jung, F. 2005. I2C bus (Inter-IC bus) [Online]
Available at: http://whatis.techtarget.com/definition/I2C-bus-Inter-IC-bus
[Accessed: 1st March 2016].

Kushner, D. 2011. The Making of Arduino [Online].
Available at: http://spectrum.ieee.org/geek-life/hands-on/the-making-of-arduino
[Accessed: 30th February 2016].

Lomas, N. 2015. Codie Joins The Ranks Of Toys Aiming To Help Kids Learn Coding [Online].
Available at: http://techcrunch.com/2015/04/03/codie/
[Accessed: 29th February 2016].

Make Wonder. 2016. Dash is a child's first real robot friend [Online].
Available at: https://www.makewonder.com/dash
[Accessed: 29th February 2016].

Mod My Pi. 2013. What's The Difference Between DC, Servo & Stepper Motors? [Online].
Available at: http://www.modmypi.com/blog/whats-the-difference-between-dc-servo-stepper-motors
[Accessed: 2nd February 2016].

Modular Circuits. H-Bridges – the Basics [Online].
Available at: http://www.modularcircuits.com/blog/articles/h-bridge-secrets/h-bridges-the-basics/
[Accessed: 4th March 2016].

Muthu, S. DCMotorBot Library [Online].
Available at: https://github.com/sudar/DCMotorBot
[Accessed: 16th March 2016]

Peterlee. 2015. Top 10 Educational Robot kit for kids [Online].
Available at: http://www.makeblock.cc/mbot/top-10-educational-robot-kit-for-kids/
[Accessed: 29th February 2016].

RakeshRon. A. 2013. L293D Motor Driver IC [Online].
Available at: http://www.rakeshmondal.info/L293D-Motor-Driver
[Accessed: 4th March 2016].

RakeshRon. B. 2014. 4 Wheeled Robot Design Basics and Challenges
Available at: http://www.rakeshmondal.info/4-Wheel-Drive-Robot-Design
[Accessed: 12th March 2016]

Random Nerd Tutorials. Complete Guide for Ultrasonic Sensor HC-SR04 [Online].
Available at: http://randomnerdtutorials.com/complete-guide-for-ultrasonic-sensor-hc-sr04/
[Accessed: 2nd March 2016].

Raspberry Pi, About Us. The Making Of The Pi [Online].
Available at: https://www.raspberrypi.org/about/
[Accessed: 3rd February 2016].

Raspberry Pi Learning Resources A. Assemble the Sense HAT [Online].
Available at: https://www.raspberrypi.org/learning/astro-pi-guide/assemble.md
[Accessed: 2nd February 2016]

Raspberry Pi Learning Resources B. Getting Started With The Sense HAT [Online].
Available at: https://www.raspberrypi.org/learning/getting-started-with-the-sense-hat/worksheet/
[Accessed: 3rd February 2016].

Raspberry Pi Learning Resources C. Software Installation [Online].
Available at: https://www.raspberrypi.org/learning/astro-pi-guide/software.md
[Accessed: 2nd February 2016]

Reddit. 2014. Interfacing RPi and Arduino via USB, TTL Serial, or I2C... what's the difference? [Online].
Available at: https://www.reddit.com/r/raspberry_pi/comments/1x2uy5/interfacing_rpi_and_arduino_via_usb_ttl_serial_or/
[Accessed: 1st February 2016].

Schwartz, K. 2014. Robots in the Classroom: What Are They Good For? [Online].
Available at: http://ww2.kqed.org/mindshift/2014/05/27/robots-in-the-classroom-what-are-they-good-for/
[Accessed: 28th March 2016].

Sharma, A. Insight - Learn the Working of Ultrasonic Sensors [Online].
Available at: http://www.engineersgarage.com/insight/how-ultrasonic-sensors-work
[Accessed: 2nd April 2016].

Tracy, 2015. OzoBot Bit Review - Mini Programmable Line-Following Robots [Online].
Available at: http://www.techagekids.com/2015/12/ozobot-bit-review-mini-programmable.html
[Accessed: 29th March 2016].

Tutorials Point A. 2016. SDLC - Waterfall Model [Online].
Available at: http://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm
[Accessed: 26th March 2016].

Tutorials Point B. 2016. SDLC - Agile Model [Online].
Available at: http://www.tutorialspoint.com/sdlc/sdlc_agile_model.htm
[Accessed: 26th March 2016].

10. Bibliography

Garage Lab. 2012. Tutorial: L293D H Bridge DC motor controller with Arduino [Online]
Available at: http://garagelab.com/profiles/blogs/tutorial-l293d-h-bridge-dc-motor-controller-with-arduino

Kawal. 2011. How to drive dc motor using L293D with Arduino [Online]
Available at: http://communityofrobots.com/tutorial/kawal/how-drive-dc-motor-using-l293d-arduino