One Shot

A fun entertaining game

PRESENTED

BY

Omar I. Shkokani, Shadi W. Basha

TO

The Department of Computer Engineering

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

Bachelor of Science (B.Sc)

IN THE SUBJECT OF

Computer Engineering

An-Najah National University

Nablus, Palestine

2023

Table of contents

Table of Figures

Disclaimer

This report was written by Omar and Shadi at the Engineering Department, Faculty of Engineering, An-Najah National University. It has not been altered or corrected, other than editorial corrections, as a result of assessment and it may contain language as well as content errors. The views expressed in it together with any outcomes and recommendations are solely those of the student(s). An-Najah National University accepts no responsibility or liability for the consequences of this report being used for a purpose other than the purpose for which it was commissioned

Abstract

Our project revolves around the development of a Dart Shooter Robot capable of operating in two distinct modes: Manual and Autonomous. While the Manual mode provides users with direct control through a mobile application, the Autonomous mode leverages camera technology and advanced sensors for precise target recognition and shooting capabilities.

In the Autonomous mode, the robot utilizes its integrated camera and sensors to autonomously identify and track targets, providing a hands-free dart shooting experience. This mode eliminates the need for manual target acquisition and enhances the user's dart-shooting accuracy.

The Manual mode, accessible through a user-friendly mobile application, remains unchanged, allowing users to take control of the robot, aims and shoot at targets using the camera's real-time vision feedback.

While previous projects may have integrated distance sensors for target detection, our project's distinctiveness lies in the incorporation of advanced camera-based visual recognition technology, eliminating the need for additional sensors and adding a layer of precision to the target acquisition process.

Our focus is on creating a versatile and engaging dart shooter robot that seamlessly transitions between manual and autonomous modes. This project amalgamates our passion for hardware innovation with the excitement of gaming, offering a unique and captivating experience for users. Our research will involve selecting suitable hardware components and optimizing the visual recognition system to ensure exceptional performance in both manual and autonomous modes, making our Dart Shooter Robot stand out in its category.

Introduction

Our project, aptly named "One Shot," emerged from a desire to combine precision engineering with the thrill of gaming. We aimed to create an enjoyable and competitive game that would captivate friends and players alike. This endeavor began with a quest for a project idea, during which we sought guidance from our mentor, Dr. Manar Qamhieh. As we delved into the realm of Arduino projects, inspiration struck when we stumbled upon a concept similar to our vision [1].

Drawing from this initial spark of inspiration, we embarked on a journey that challenged our knowledge and put into practice the concepts we had acquired along the way. The project presented us with formidable challenges, primarily centered on the integration and communication between various controllers. Additionally, we grappled with the intricacies of image processing, further complicated by environmental factors.

"One Shot" is a project that encompasses a diverse array of concepts. Notably, it involves the integration of controllers through serial communication between an Arduino and ESP8266, the exchange of HTTP packets between a Raspberry Pi and the ESP8266, as well as HTTP communication between the Raspberry Pi and the Blynk mobile application. Furthermore, the project encompasses the domain of image processing and the precise control of servomotors.

Our passion for gaming and the joy of playing propelled us towards this idea. While we designed "One Shot" with gaming enthusiasts in mind, it is worth noting that the nature of the project, particularly the inclusion of nails on the arrows for target retention, makes it suitable for teenagers and young adults rather than younger children.

Our journey began with a review of the design available in the Arduino projects, where we discovered that a similar project had been previously executed [2]. However, we borrowed only the 3D printed model from the previous iteration. The bulk of our effort went into planning and implementing distinctive features that would set our project apart. Once we had outlined these features, we proceeded to procure the necessary hardware components, ranging from controllers to motors. The subsequent phase involved rigorous testing, ensuring that each component performed as expected. Finally, we brought all the elements together and focused on fine-tuning the project's functionality, resulting in the creation of "One Shot."

Literature Review

Our project's inception was rooted in inspiration drawn from an Arduino Project Hub endeavor [1]. We adopted this idea as a foundation for our own, seeking ways to incorporate our hardware knowledge and push the boundaries of innovation. To gain insights and improve upon the concept, we reviewed a similar project undertaken in a previous semester [2]. This review provided invaluable lessons on what pitfalls to avoid and what enhancements we could introduce to create an improved and more sophisticated version of the project.

One glaring limitation in the prior project was the absence of image processing capabilities and a restricted range of movement, particularly the inability for 360-degree rotation. Building upon these observations, we set out to address these shortcomings and introduce novel features. Notably, we implemented a point system that capitalized on image processing, adding a layer of engagement to the project.

Diverging from the previous project's heavy reliance on multiple distance sensors for both movement and target detection, we adopted a more advanced approach. We integrated a camera system to facilitate visual recognition of the surrounding environment. This transition eliminated the need for an array of sensors, streamlining our design while enhancing precision.

The integration of a Raspberry Pi for image processing and target detection represented a significant leap forward in our project's capabilities. We delved into the powerful OpenCV library to refine our target recognition, resulting in a more sophisticated and efficient system. This integration also necessitated a deeper understanding of the communication protocols between the Raspberry Pi and our primary controller, the ESP8266. We harnessed HTTP packets for seamless data exchange, achieving a robust integration.

Furthermore, we encountered a voltage mismatch issue during the project. The ESP8266 operated at 3.3V, while the motors required 5V signals. To resolve this, we explored solutions and discovered the practicality of level converters. These converters effectively bridged the voltage gap, ensuring compatibility between our components and enabling precise motor control.

In addition to these challenges, we ventured into the realm of image processing to tackle the problem of target detection, ultimately enhancing our robot's precision in locating and engaging targets. To overcome the fundamental challenge of achieving 360-degree rotation capabilities, we devised an innovative solution involving the integration of IR sensors, which enabled seamless and responsive movement in all directions.

In conclusion, our literature review displays the project's evolution, underpinned by a commitment to hardware innovation, continuous learning, and problem solving. By drawing upon experiences and addressing inherent limitations, we have transformed a rudimentary concept into a sophisticated Dart Shooter Robot that offers users an engaging and immersive experience.

Methodology

The project as mentioned before is a small robot, its structure is a premade 3D model; the model is available in the Arduino project hub, and contains many features mainly:

- Manual mode, using the app or the screen the user can control the robot to hit the target.
- Auto mode, using image processing the system detect the target and hits it.

To do that, we had to implement multiple side features, which are:

- System of communication between different controllers.
- Blynk app that sends commands for the controller to move that robot.
- Controlling a variety of hardware components to achieve our goal.
- Image processing to get the target and compute the points.

Constraints

Limited Presence of Black Circles

The environment where the robot operates must not contain any black circular objects other than the designated target. This constraint ensures that the image processing algorithms can accurately distinguish the target from the background.

Limited Range of Servomotor Rotation

The 360-degree servomotor has physical limitations due to wiring constraints. It can rotate up to 180 degrees to the right and 180 degrees to the left, resulting in a blind spot directly behind the robot. This constraint necessitates careful positioning to ensure that the target is within the motor's range of motion.

Lighting Conditions for IR Sensors

The IR sensors require appropriate lighting conditions to operate smoothly. Excessive or insufficient lighting can affect their functionality. Maintaining optimal lighting levels within a specified range is crucial to ensure accurate obstacle detection.

Target Compatibility

The system is designed to work exclusively with the predefined target. Compatibility with other targets or objects is not guaranteed, as the image processing algorithms and targeting mechanisms are tailored specifically to the unique characteristics of the provided target.

Limited Operating Range

The effective operating range of the robot is constrained to a distance between 1 and 2 meters from the target. Operating the robot beyond this range may result in reduced accuracy and effectiveness in target detection and dart shooting.

Hardware component

Our system contains a variety of hardware:

1. ESP8266mod Node MCU 1.0
2. Arduino Uno
3. Raspberry Pi 3
4. Raspberry Pi camera V 1.3
5. shield touch LCD TFT 2.8 inch
6. 3 servo motors: details later
7. two IR sensor
8. two dc-motors
9. one channel relay
10. two regulators 7806
11. l298n driver
12. laser
13. two logic level convertor
14. capacitors
15. bread board
16. power supply
17. wires

We will go through each component and why did we use it starting from the ESP8266:

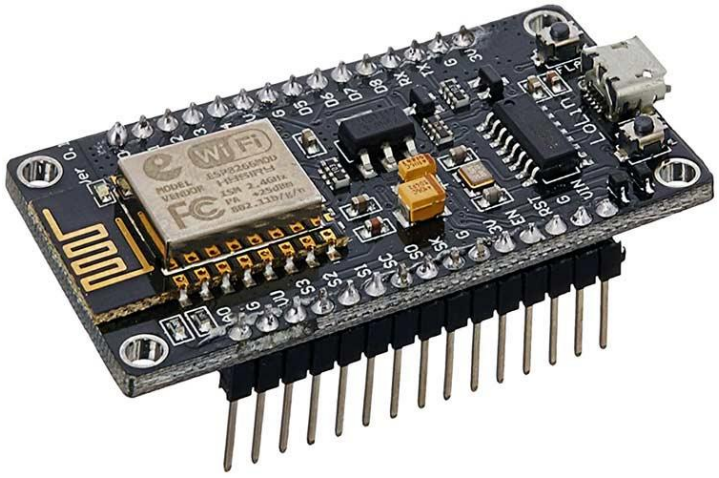

Figure 2 - ESP8266mod Node MCU 1.0

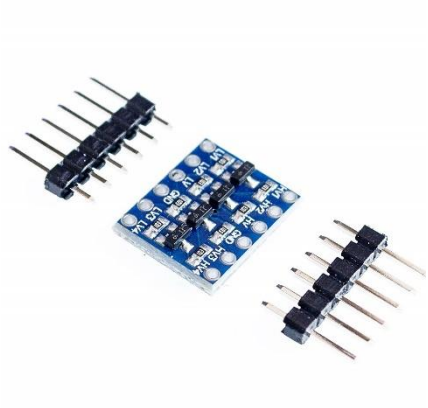

Figure 1 - logic level converters

Our main controller is the ESP8266 (Figure 2) because it provides a one UART, which is used to send commands between the ESP8266 and the Arduino Uno, and it can be connected to the Wi-Fi, so we used it to communicate with the Raspberry Pi 3 and the Blynk application, moreover it provides 17 GPIO to control the system.

However the we faced a problem which is that the output of the ESP8266 is 3.3 volts and to control the motors or to communicate with the Arduino using the serial we needed 5 volts so we used the logic level converters (Figure 1) to convert the inputs to the ESP8266 to 3.3volts and the outputs to 5 volts.

The ESP8266 is responsible for controlling the Motors, our set of motors were

- Two DC motors (Figure 6)
- 180-degree plastic servomotor (Figure 5)
- 180-degree metal servomotor (Figure 4)
- 360-degree metal servomotor (Figure 3)

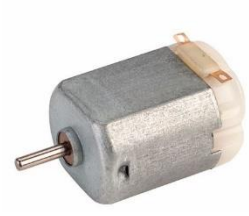

Figure 6 - DC motor

Figure 5 - 180-degree plastic servomotor

Figure 4 - 180-degree metal servomotor

Figure 3 - 360-degree metal servomotor

The dc motors were used for their high RPM to through the arrow to the target, and the plastic servomotor was to act like a trigger to push the arrow. We tried the plastic servomotor for the vertical movement but it did not work because it could not lift the model, so we got the meatal servo motor to work for the vertical movement, and lastly we used a continuous 360-degree servomotor to be able to rotate the robot in 360-degree horizontally.

However, we faced a problem on the 180-degree servomotor that works for the vertical movement and the 360-degree servomotor for the horizontal movement, both of the motors were weak and working on a low torque, so we had to use a 7806 regulator (Figure 7) to provide the motors with 6 volts to work at their best.

Figure 7 - 7806 regulator

In addition, the ESP8266 uses a relay (Figure 10) to control the laser (Figure 8), the relay is used to make the laser work on 5 volts and be controlled by the ESP8266, and the laser is used to indicate an approximation of where the shot will land.

Figure 10 - relay

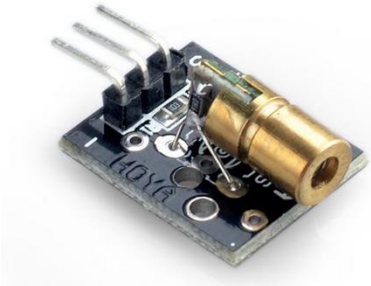

Figure 8 - laser

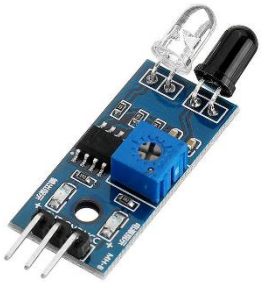

Figure 9 - IR sensor

The ESP8266 takes input from two IR sensors (Figure 9), one is used to detect if the model has no arrows in the magazine and the second one is used to detect if the model rotated to the zero angel or the 360 angel to insure that the cablese are not effected by twisting them.

All of the system is connected to the power supply (Figure 11), which provides 12 volt and 12.5A, but we need to control the DC motors so we used the 1298n driver (Figure 12), which provides two outputs for 12 volts and 5 volts. Therefore, the DC motors are connected to the driver on the 12 volts and the ESP8266, laser and the plastic 180-degree servomotor (trigger servomotor) is connected to the driver on the 5 volts.

Figure 11 - power supply

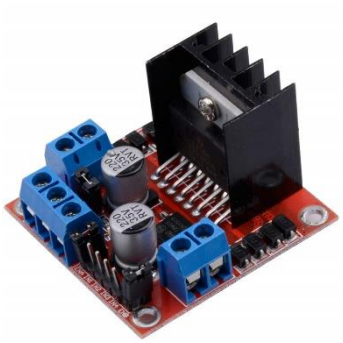

Figure 12 - 1298n driver

The next component is the Arduino Uno (Figure 14); we used the Arduino to connect with the Screen (2.8″ TFT Touch LCD Shield) (Figure 13) and it is connected with the ESP8266 through the serial. Therefore, the Arduino will give the project the user interface to interact with the project.

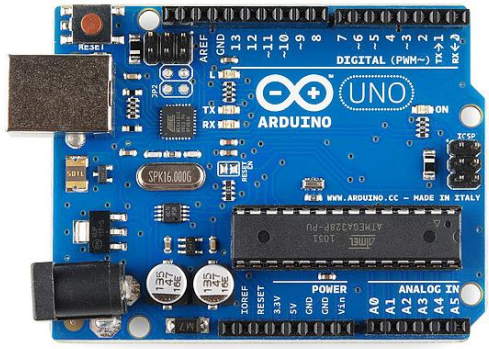

Figure 14 - Arduino Uno

Figure 13 - 2.8 TFT Touch LCD Shield

The last controller that we have is the Raspberry Pi 3 (Figure 15), which is responsible for image processing for the tracking and the point calculation and its connected to the system with the ESP8266 using the Wi-Fi; it is attached to the Raspberry Pi camera V1.3 (Figure 16) to get the images for the processing.

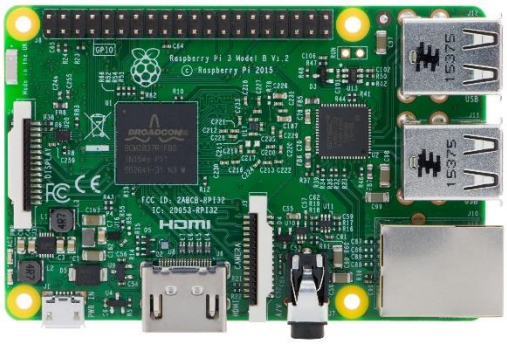

Figure 15 - Raspberry Pi 3

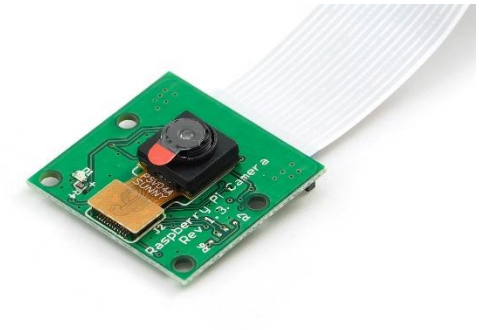

Figure 16 - Raspberry Pi camera V1.3

Physical connections

We will start taking about the communication between the system and how it is all connected together to achieve the goal and make.

The figure (Figure 17) below shows how our system is connected

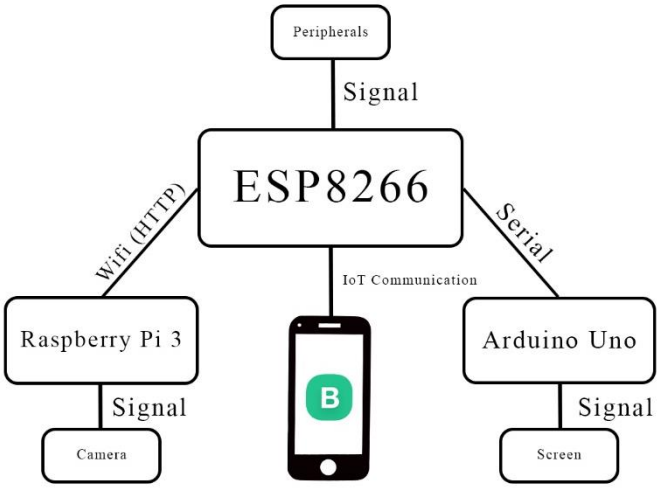

Figure 17 - System connectivity

The figures below shows all the connections.

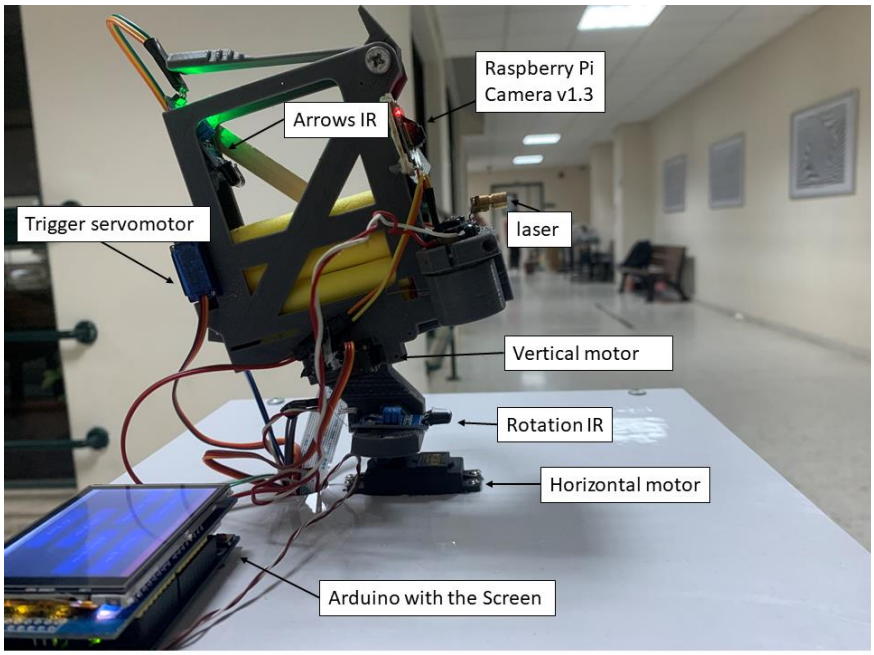

Figure 18 - side view

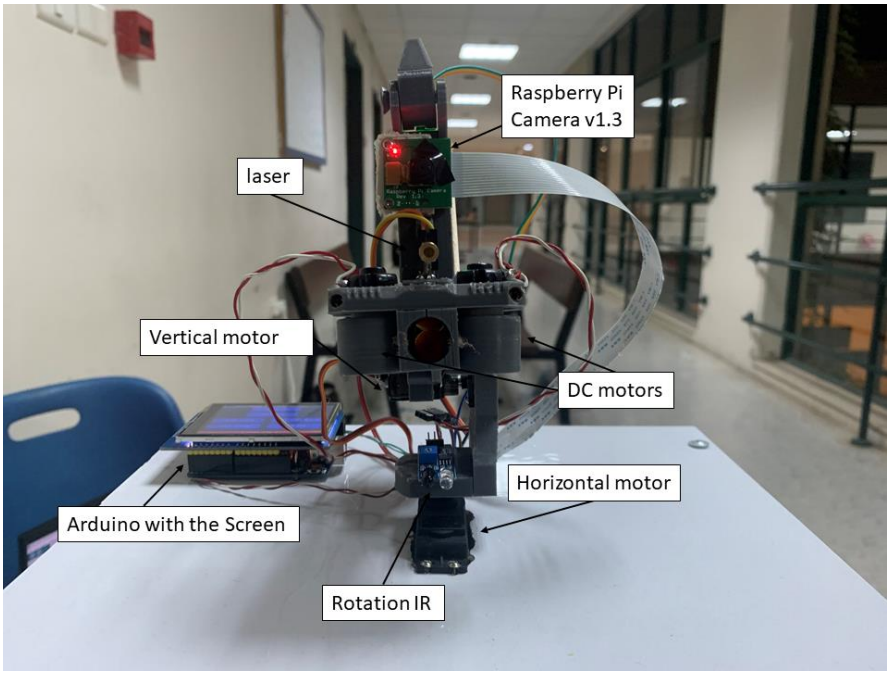

Figure 19 - front view

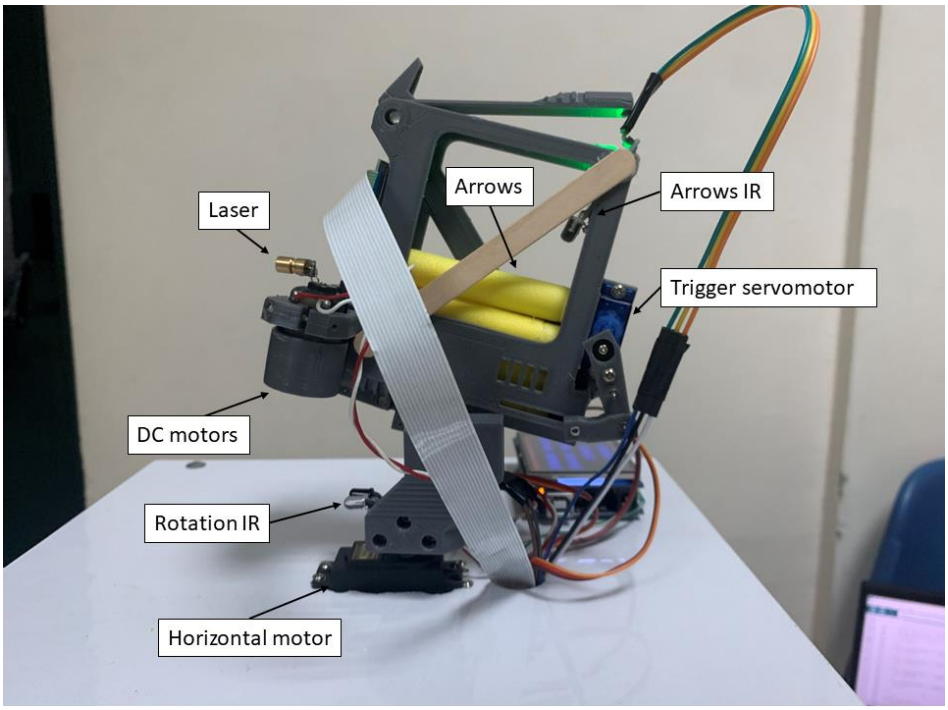

Figure 20 - side view

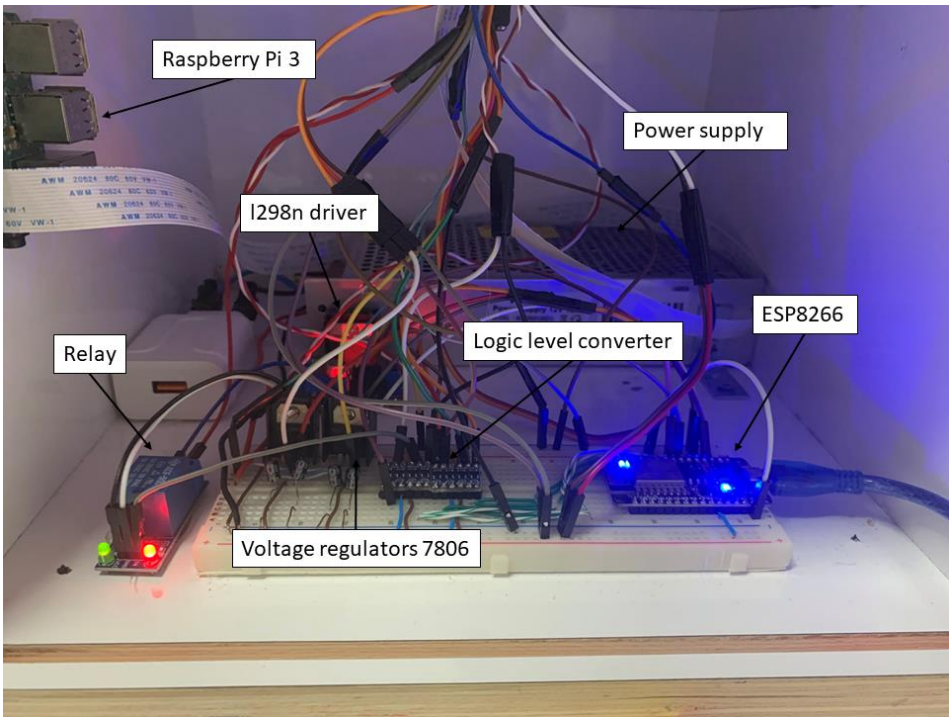

Figure 21 - connections

Here is the full blueprint for the system.

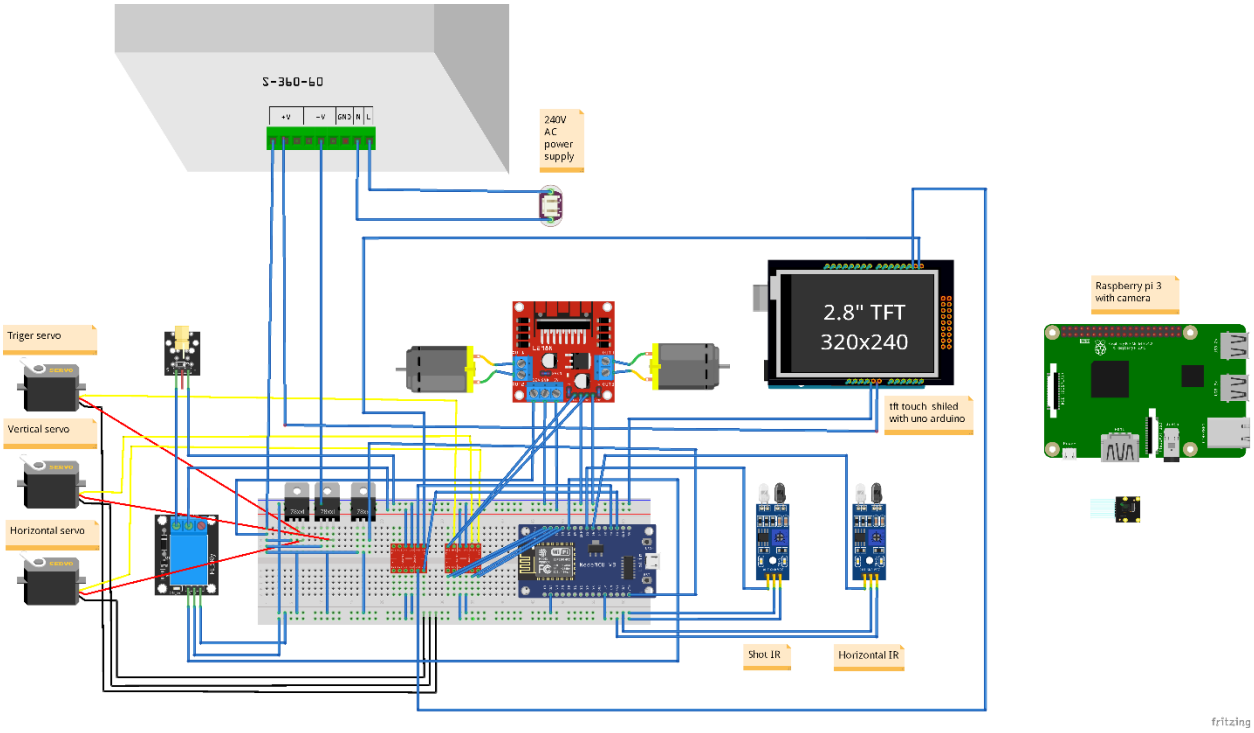

Figure 22 - full connection

We will start with the basic communication between the ESP8266 and the motors, laser, and the IR sensors.

ESP8266 with peripherals

The ESP8266 is connected to three servomotors, two IR sensors, and a laser. The laser and the motors are connected to the ESP8266 through a logic level converter because the signal in motors and the relay requires 5 volt input range, and the ESP8266 provides only a 3.3 volts input range. However, the IR sensors can operate on 3.3 volts, so they are connected directly to the ESP8266 pins and they take voltage from it too. Unlike the motors, they take their power form the regulators, except the trigger motor it takes the power with the laser from the l298n driver as shown in the figure.

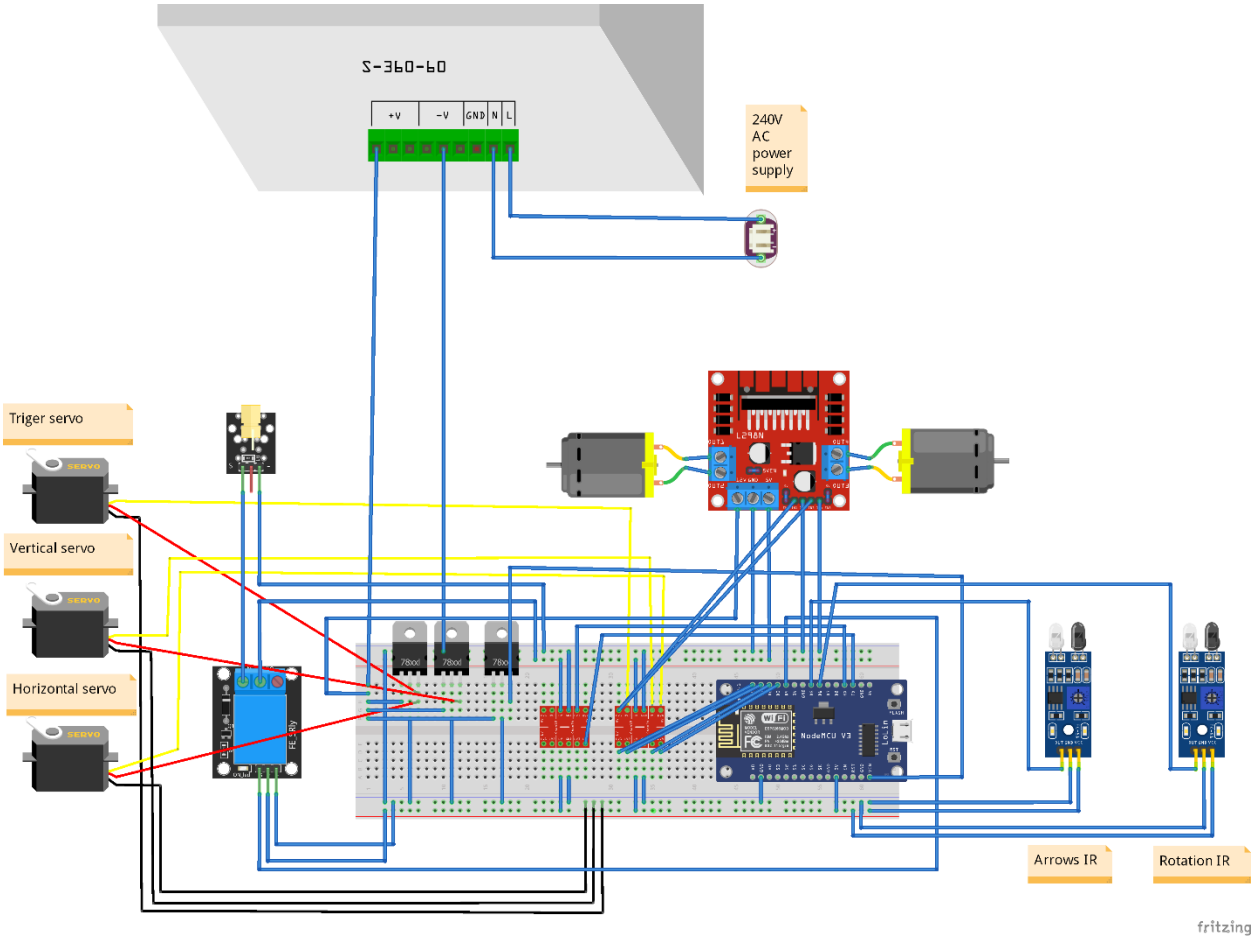

Figure 23- ESP8266 with peripherals

ESP8266 with the Arduino

The Arduino and the ESP8266 are connected together through the serial pins (TX/RX), however they are not directly connected, the RX in the ESP8266 cannot be connected with the TX from the Arduino because the Arduino gives an output range from 0 to 5 volts and the input range for the ESP8266 is 0 to 3.3 volts. Therefore, to insure the safety of the components we connected them through a logic level converter.

In the Arduino, the input range from 0 to 5 volts taking 2.7 volts as high logic so there is no problem connecting the TX from the ESP8266 to the RX in the Arduino directly, but we connected them through a logic level converter just to make sure if a noise hits the signal don't get affected.

The Arduino is powered directly from a 12-volt source. The figure below shows the connections.

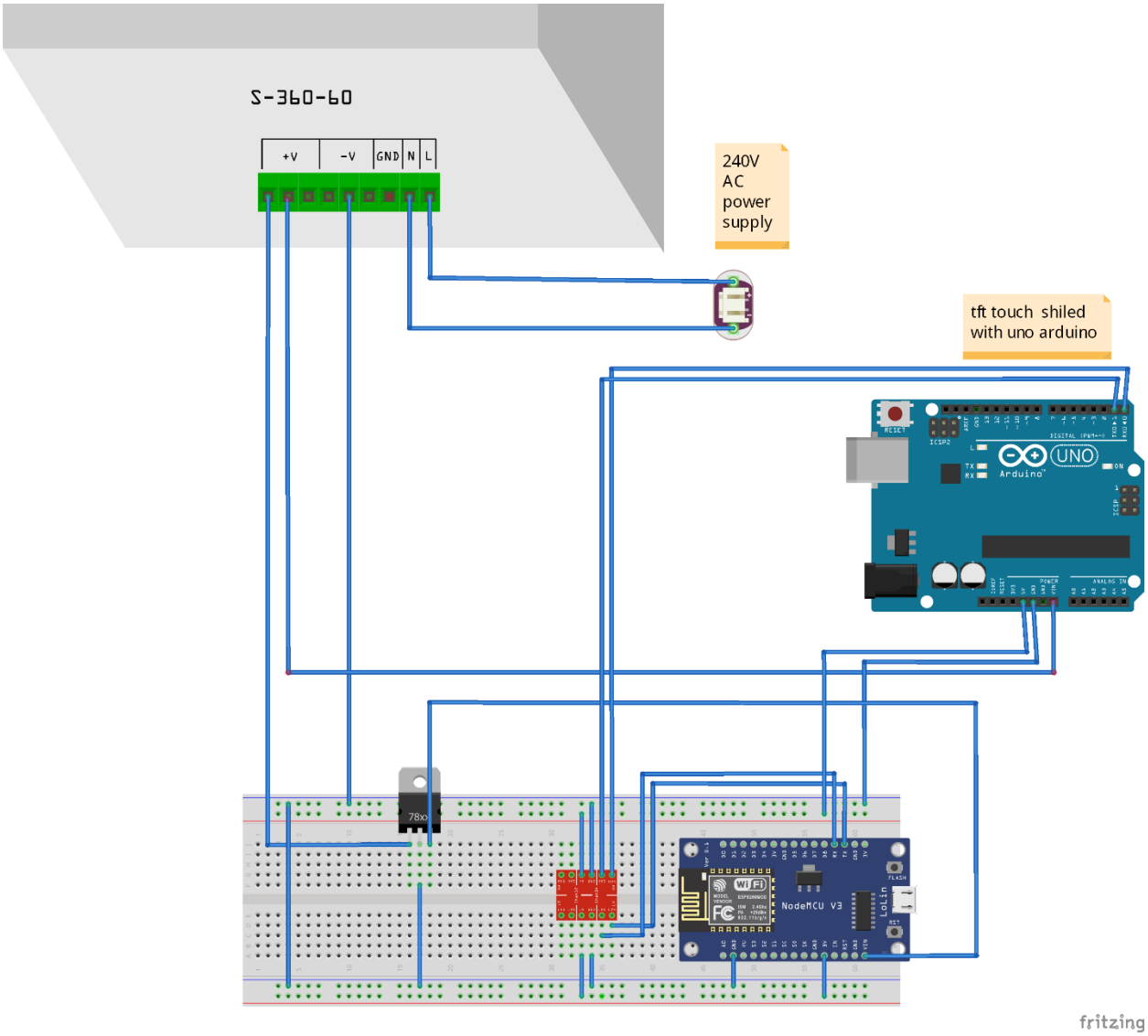

Figure 24 - ESP8266 with the Arduino

Arduino with the Screen

The Arduino is connected to the Screen directly through the shield, and serially with the ESP8266 using the TX and RX pins. The figure below shows the connections.

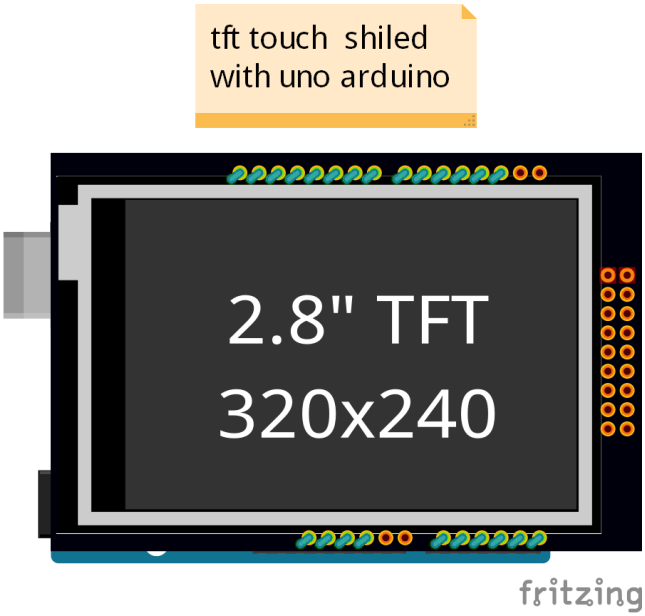

Figure 25- Arduino with the Screen

Raspberry Pi with the camera

The Raspberry Pi is not connected physically with the system it is just connected to the camera through the flex cable. The figure below shows the connections.

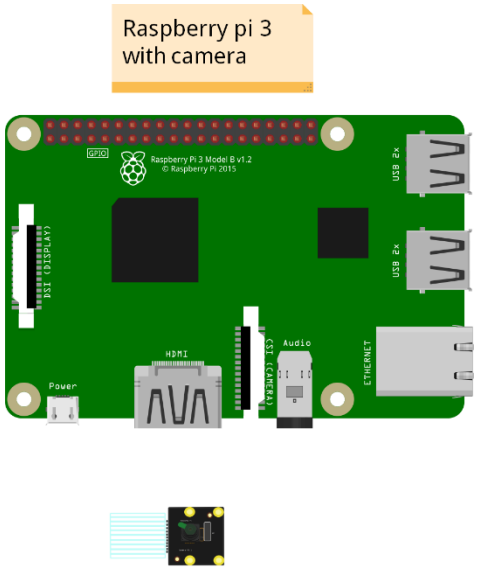

Figure 26 - Raspberry Pi with the camera

System logic

The system works in two modes as mentioned before a Manual Mode, and Robot Mode. Both modes have a common ground on mechanical movement and sensor readings, So we will first go through each of the basic functionality then we will move to the modes.

Basic functionality

The robot have four deferent mechanical movements, which are up, down, left, and right, all of those are done be the vertical and horizontal motors. The figure below shows how to model moves in the four directions.

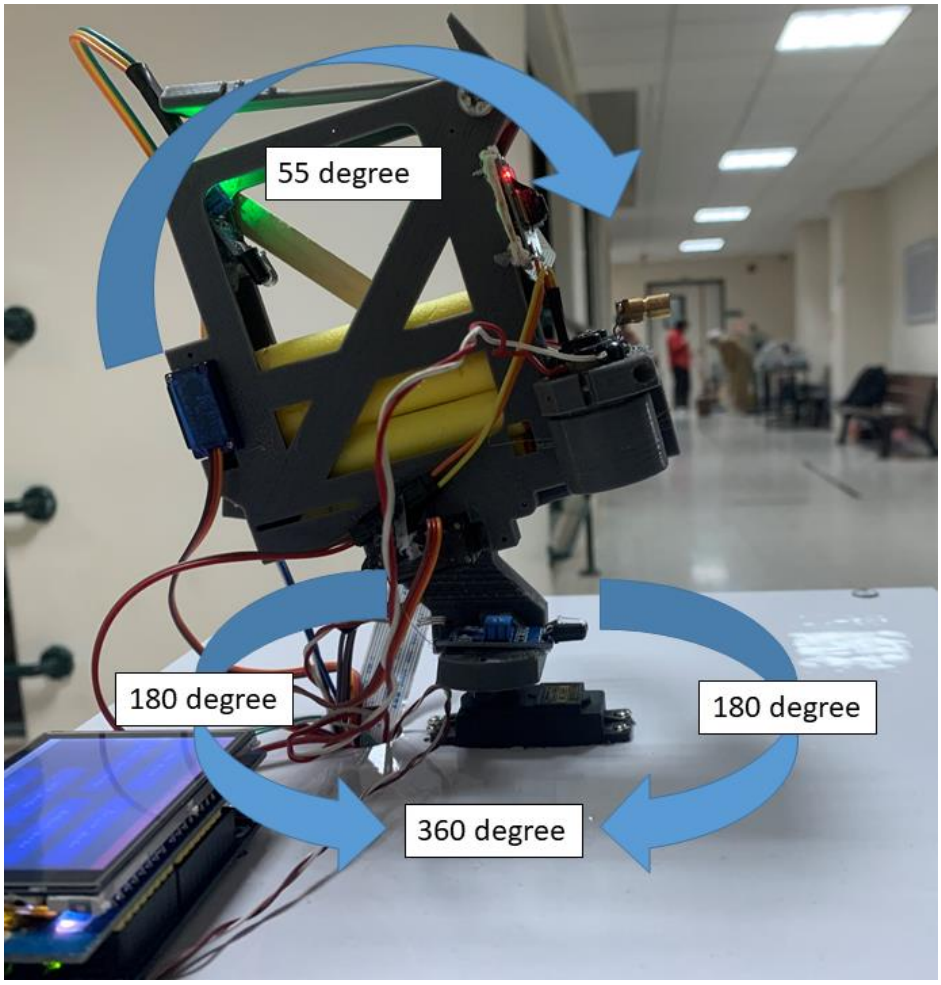

Figure 27- horizontal and vertical movement

As shown in the figure 27 the robot can rotate in approximately 360-degrees, and vertically 55-degrees. We limited the horizontal to 360-degree by using the rotation IR sensor, when it lights that means it hits the wires so the robot is above 360-degree or below 0-degree.

At first, we wanted to add a stepper motor for the horizontal movement but it does not fit in the 3D printed model so we used the 360-degree servomotor. However, another problem popped up that we cannot control the horizontal servomotor because it is continuous, and balancing the right and the left movement is not a choice because the left movement is stronger than the right one, so we used the IR sensor.

The vertical servomotor is limited by the code to 55-degrees because the robot above that will cause a pressure on the wire and below these 55-degrees, the arrow will hit the box.

The next functionality is the shot; the mechanism used in shooting the arrows is moving the DC motors in opposite directions (one counter clock wise, two clock wise) is starts for a short period of time to let them gain some speed after that the trigger motor will move the trigger which will push the arrow to the motors. The motors then will catch the arrow and push it outside. During the shot period, we turn off the laser because it vibrates; the fast movement in the DC motors causes this vibration, the figure below shows the steps of the shooting.

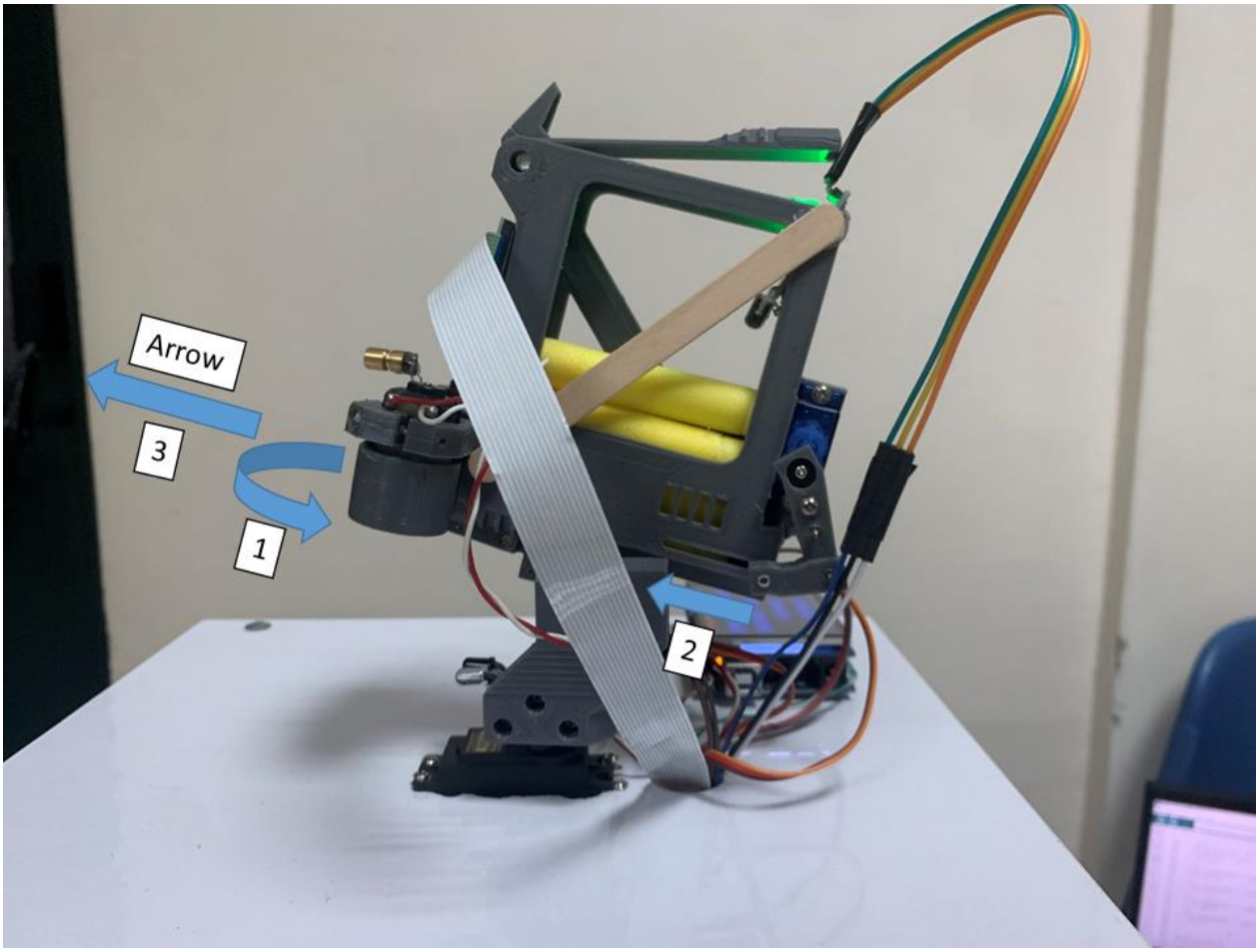

Figure 28 - Shooting mechanism

However, the shot cannot function without having an arrow in the magazine, the arrows IR sensor detects if there is arrows in it, and it will shoot other than that it will not shoot.

The basic functionalities are implemented in five functions:

- Go up: this function moves the vertical servomotor one degree up and checks if the move exceeded the limit if so it does not commit to it.
- Go down: this function moves the vertical servomotor one degree down and checks if the move exceeded the limit if so it does not commit to it.
- Go left: this function moves the horizontal servomotor one step to the left and check the IR for the 360 movement.
- Go right: this function moves the horizontal servomotor one step to the right and check the IR for the 360 movement.
- Shot: this function turn off the laser and turn on the DC motor then waits for 1.5 sec for the DCs to speed up, then moves the trigger servomotor to shoot the arrow.

Image processing

As mentioned before we have used the Raspberry Pi for the image processing, to detect the target. We have used the Hough Gradient Method to detect the circles because it has a better ability to handle partial occlusion, and capability to work with non-uniform intensity variations. It can also accurately determine circle centers and sizes.

The Raspberry pi when first the initiated the it opens a service on port 8888, and work as a server, then it waits for the ESP to connect with her, after connecting with the ESP it waits for a command called “track” this command do the following:

It detects the target, shown in the figure below. By getting, the black circle because it is the best option and the easiest to detect using the Hough Gradient Method. This method brings back the center of the circles that are black, so we take the largest one and send the center and the radius to the ESP, if no circles are detected It returns (-1, -1, -1) (x, y, radius).

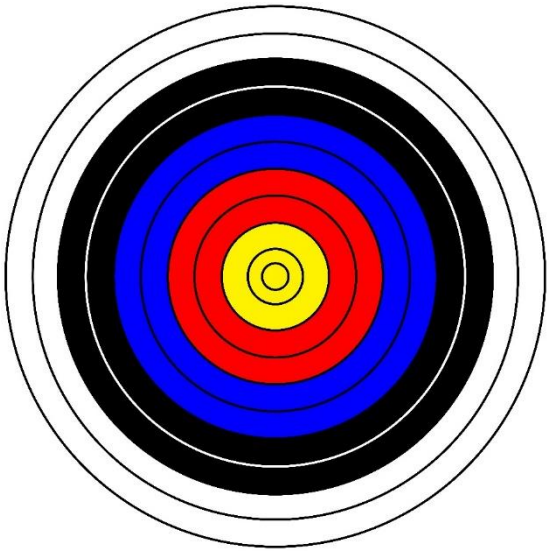

Figure 29 - The target

Manual mode

This mode allows the user to control the robot using the Screen or the Blynk application, the user chooses one of the manual modes app mode or LCD mode as shown in the figure below. In general when LCD handles a press action on any button, the Arduino sends a command using serial communication to tell the ESP to change the mode, the Arduino Uno then remains in a waiting state until it receives an acknowledgment from the ESP, confirming the correct reception of the command.

The app mode connects the ESP8266 with the Blynk cloud via Wi-Fi using a hardcoded key, which allows it to be controlled by the app that we designed (figure 31)

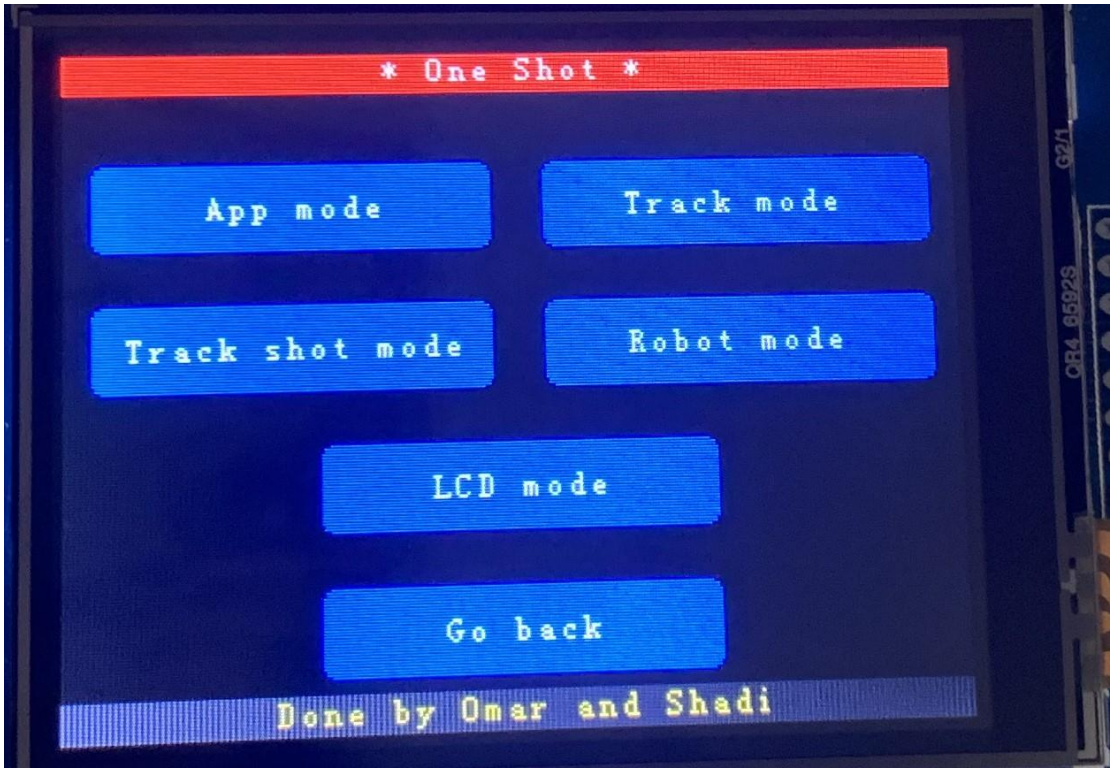

Figure 30 - Modes screen

Each button in the app triggers a command in ESP; the up calls the up command, the left calls the left command and so on.

The same for the LCD mode the each button is responsible to trigger a function of up, down, left, right, and shoot and u can exit the mode using the go back button

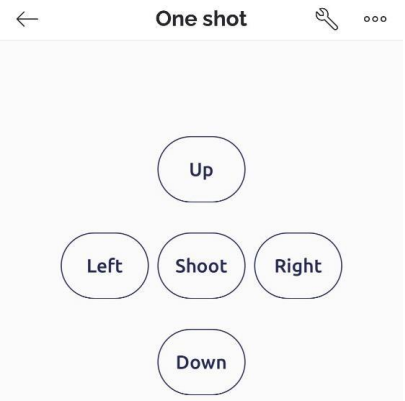

Figure 31 - Blynk application

Figure 32 - LCD mode

Auto mode

This mode is divided into three different modes track mode, track and shot, and robot mode, each one of them depends on the communication between the ESP and the Raspberry Pi, the Raspberry Pi waits for a track command from the ESP to detect the target and send the coordination of the center through an HTTP packet.

When the mode is set to tracking, the ESP sends the track command to the Raspberry Pi, and then the Raspberry Pi responds with the center of the target if it exists. If the target does not exist, it sends (-1, -1, -1). Based on that information, the ESP decides which way it should go to follow the target within the range of the yellow circle. Initially, if the target is not found, the ESP operates in scanning mode, where it keeps rotating to the maximum left and then to the maximum right until it finds the target. If it finds the target, the ESP keeps tracking it.

The next mode is track and shot, in this mode, it tracks the target until it reaches the yellow circle, when there is no more movement and the target is steady it shoots at it.

The last mode is robot mode. In this mode, the robot tracks and shoots at the board until the arrows are empty, depending on the IR sensor.

Arduino Commands to ESP8266

The Arduino is the main controlling unit for changing the commands, it sends command lines to the ESP8266 through the serial. We have built a set of commands to change the modes in the ESP8266 we have three main commands categories:

- "change_mode": this command states that the next line form the Arduino will be one of the modes using the commands:
    - "app_mode": Enters the App mode
    - "track_mode": Enters the Track mode
    - "track_shot_mode": Enters the Track and shot mode
    - "robot_mode": Enters the Robot mode
    - "lcd_mode": Enters the LCD mode
    - "off_mode": Exits form the current mode
- "lcd_mode": this command works when the screen is on the LCD mode, and states that the next command form the Arduino will be one of these commands:
    - "up": runs the going up function
    - "down": runs the down function
    - "right": runs the right function
    - "left": runs the left function
    - "shot": runs the shot function

Conclusion

The journey of creating "One Shot" has been both challenging and rewarding, resulting in the development of a captivating dart shooter game that merges hardware innovation with the excitement of gaming. This project, born from inspiration found in Arduino projects and guided by the mentorship of Dr. Manar Qamhieh, encapsulates our dedication to pushing the boundaries of what is possible with hardware integration and image processing.

We have gained invaluable insights into the intricacies of integrating diverse controllers, mastering power management, and delving deep into the world of image processing.

Limitations:

Our journey encountered several challenges, particularly in the realms of accuracy and image processing. Achieving precision in dart shooting proved to be a formidable obstacle, as did ensuring the seamless operation of the IR sensor, which necessitates a controlled lighting environment for optimal functionality.

Future Work:

Looking ahead, there are several avenues for further development and enhancement of "One Shot." These include:

Enhancing Accuracy: Our primary goal is to improve accuracy by narrowing the target range and achieving pinpoint accuracy around the laser's focal point.

Safety Improvements: a wider age range, including children, can enjoy exploring alternative arrow designs or mechanisms to ensure the game.

Enhanced Image Processing: Investigating advanced image processing algorithms and hardware modifications to improve target recognition under varying environmental conditions.

Multiplayer Features: Expanding the game's capabilities to accommodate multiplayer modes, enabling more competitive and social gameplay experiences.

Scoring System: Refining and expanding the point system to provide players with a more comprehensive and rewarding scoring mechanism.