Mobile Robotic Arm

Naman Tanwar (20BEC0400)
School of Electronics Engineering
Vellore Institute of Technology
Vellore, Tamil Nadu, India
naman.tanwar2020@vitstudent.ac.in

***Abstract*—Exploration of various unreachable/hazardous places is a key to scientifically enrich our understanding of this universe. And remotely performing operations on these locations opens more aspects of not even exploration but also providing medical aid where a real human intervention is not possible. This project aims to serve this goal of remote exploration and operation.**

***Keywords*—Robotic Arm, Mobile vehicle, Remotely Operated, ESP32, Raspberry Pi, Servo Motor, Blynk, Cloud service**

I. Introduction

There are many places where human intervention is not possible simply because it might cost a human life. The requirement of intervening those places might be to explore those parts of the world, to fix some critical faults at places where human intervention is very dangerous, to save another human life by providing medical aid, or military reasons like spying on terrorist operations.

Hence to serve these purposes, this project aims to build a remotely controlled robot which can operate on such remote places.

This project basically is a robotic arm mounted on a movable chassis. Ultrasonic sensors are used to provide the sense of direction and a camera is used to visually inspect the target on which the robot will operate. The whole construction can be controlled remotely via internet. The only requirement is that that place should have a mobile internet coverage. This can also affect the places it can operate but different modes of connectivity can be used according to the location of operation.

II. Design And Implementaion

A. Methodology

The working of the project can be defined as:

- 2 *Ultrasonic sensors* are used to measure the distance from any obstacle in any of the four directions.
- One of the *Ultrasonic sensors* is mounted on a *servo motor* which will help to measure the distance from a obstacle in 3 directions using only one sensor
- Wheels and the robot arms are controlled using *ESP32* microcontroller
- A live camera is setup using *Raspberry Pi* which can be accessed remotely from any location. This makes the Robot perfect for operations in remote locations.
- The robot can be controlled remotely using *blynk IoT Platform*

Temporarily, a 12v SMPS is used instead of a LiPo battery to power the robot.

B. Mechanical Design and Implementaion

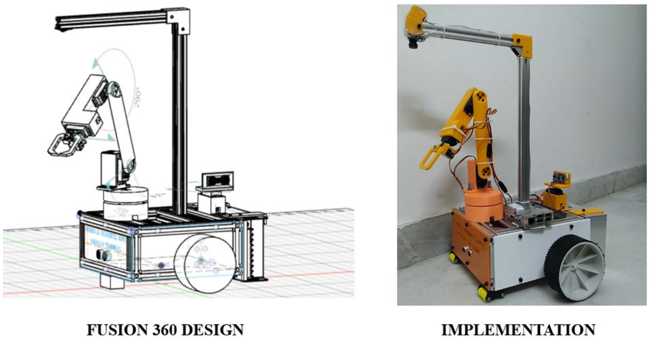

Fig. 1. Mechanical Design and implementation

The complete mechanical structure is designed using *Fusion 360* keeping in mind the real word parameters such as servo motor size, motor diameter, wheel size, circuit board dimensions, power supply dimensions, centre of movement, etc. About 14 individual parts were designed which then was 3d printed.

The robotic arm parts, sensor mount, PCB mount was 3D printed with PLA filament.

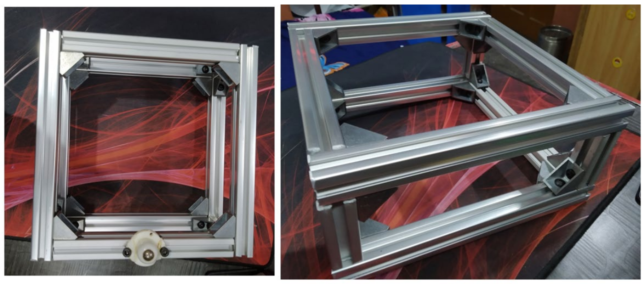

Fig. 2. Robot Chasis

The robot chassis was made using *2020 Aluminium Extrusion profile*. The profile was cut with an angle grinder according to the 3D design then joined together with lead-based corner brackets. The mounts for the camera were also made using this aluminium profile.

The Orange and white colour acrylic sheet (as seen in *Fig. 1*) is also designed using *Fusion 360* then laser cut through an online service (www.Robu.in) to mount the front sensor and *DC geared motors*. [3]

The wheel diameter is 100mm which lifted the whole chassis by 22mm and hence about 22mm castor wheels were used in front and back to provide stability to the robot

C. Circuit Diagram, Components used and circuit:

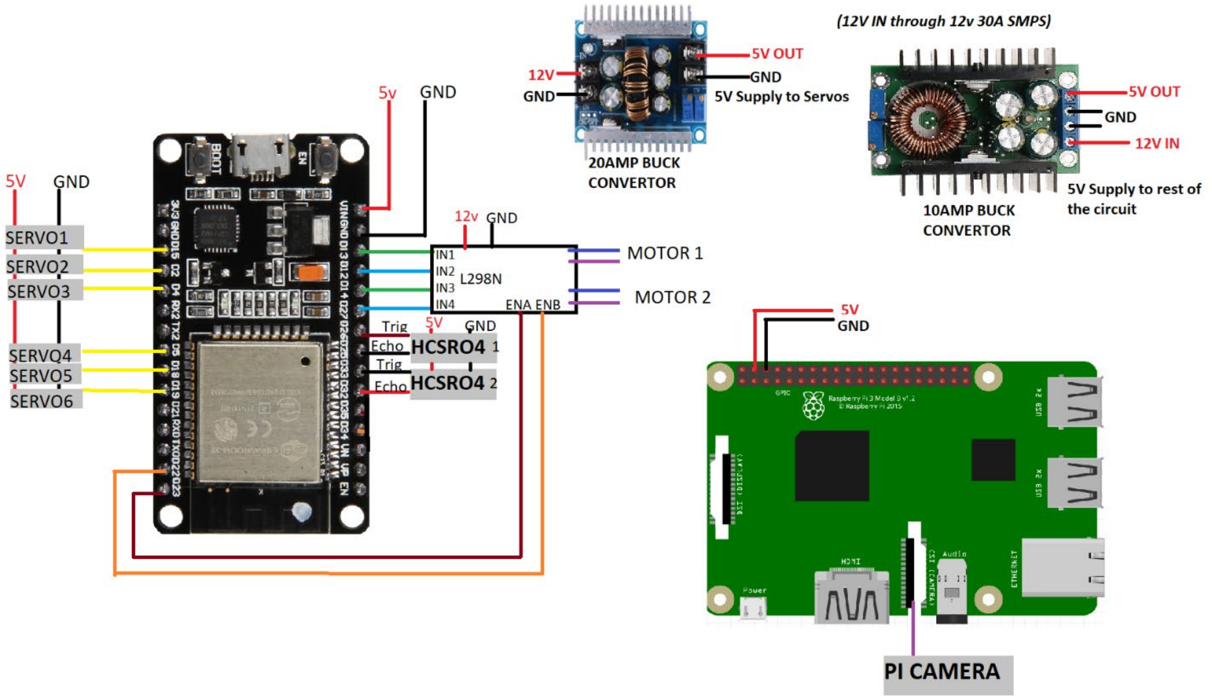

Fig. 3. Circuit Diagram

The Circuit has three parts:

- Power Supply
- ESP32 and connected devices
- Raspberry Pi and camera

1. Power Supply

A *12V 30A rated SMPS* is used to supply power to the entire system. Sisce this power supply is temporary, It can be replaced with a suitable LiPo battery (preferably, *3S 35C 3500MAh*) for mobility.

The Volatage is then step down to 5v using *Buck Covertors* so that it can be used to power other components.

The servo (*Tower Pro MG995*) can pull upto 1.5A to 2A of current under stall condition. So all the 6 servers can pull upto 12A of current. Hence, a separate buck converter (rated 20A) is used to power servos alone. The remaining buck convertor (rated 10A) is used to power rest of the circuit.

2. ESP32 and connected devices

ESP32 is used as a microcontroller as it has WiFi functionality which can be used to connect to the internet.

Six *Tower Pro MG995* servos are connected to the digital pins of ESP32 Microcontroller (D2, D4, D5, D15, D18, D19). L298N Motor Driver is used to control two *200 RPM geared DC motor*. The inputs are connected to D12, D13, D14, D27 while enable is connected to D23 and D22. Two *HCSRO4 Ultrasonic senosrs* are used to measure distance in all four direction. Digital Pins D25, D26, D32, D33 are used for connections of ultrasonic sensors.

A Raspberry Pi camera is connected to Raspberry Pi 3B+ which can be used to view camera preview remotely.

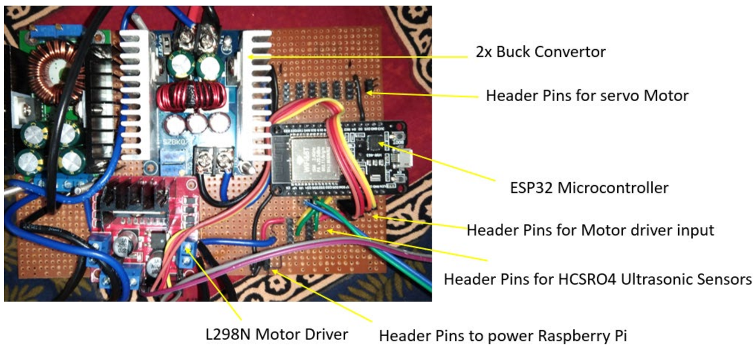

Fig. 4. Circuit

The circuit is made on a PCB board. *Male header pins* are used to connect the servo motors, Ultrasonic sensors, Motor Driver input and power input for Raspberry Pi. *Femal headers* are used to mount ESP 32 on the PCB.

In the top left, two Buck convertors can be seen mounted with some screws on the PCB.In the bottom, L298N motor driver is mounted with some screws.

D. Raspberry Pi Camera setup:

- A web interface is setup for the Raspberry Pi camera through a Open Source GitHub Project. But this can only be viewed locally. *[2]*
- Then, using remote.it platform, the local webpage is streamed to the internet which can be accessed through a specific URL. [4]
- This URL is then used in the Blynk IoT platform to watch the video stream through the mobile app. [5]

E. Servo Calibration

- The MG995 servos used for the arm was not calibrated to move to a correct angle.
- For an input of 0 to 147 degrees (using Arduino IDE) the servo moved from 0 to 180 degrees. So, the servos were calibrated accordingly.
- The fully open and fully close point for the servo used in the claw was found out to be 180 and 140 respectively

F. Movement Equations

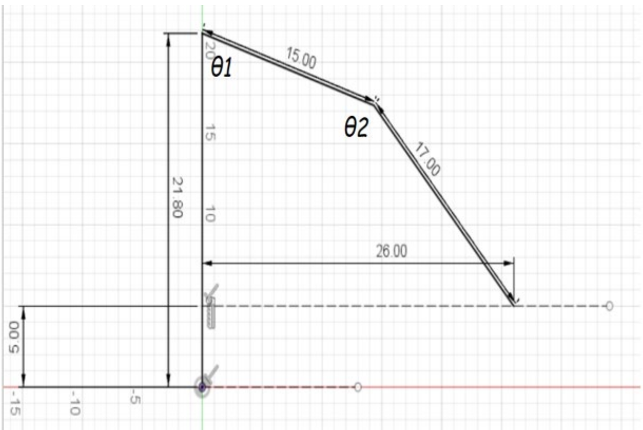

Fig. 5. 2D constrained sketch in Fusion 360

A *2D constrained line schemetic* was setup in Fusion 360. For different vallues of the distance, angle values was found out which then was inserted into Excel to find a polynomical equations between the distance and the angle

For θ1

- If d<15

$y = -0.00005x^5 + 0.0026x^4 - 0.0495x^3 + 0.324x^2 + 1.6995x + 67.123$

- If d>15

$y = -0.0002x^6 + 0.0207x^5 - 1.0631x^4 + 28.989x^3 - 442.53x^2 + 3585.8x - 11957$

Fig. 6. Movement Equations (1)

For θ2

$y = 0.000007x^6 - 0.0006x^5 + 0.0189x^4 - 0.3108x^3 + 2.8174x^2 - 11.677x + 82.814$

Here,

x = d

y = θ1 or θ2

Fig. 7. Movement Equations (2)

Fig. 8.

G. Mobile App using blynk

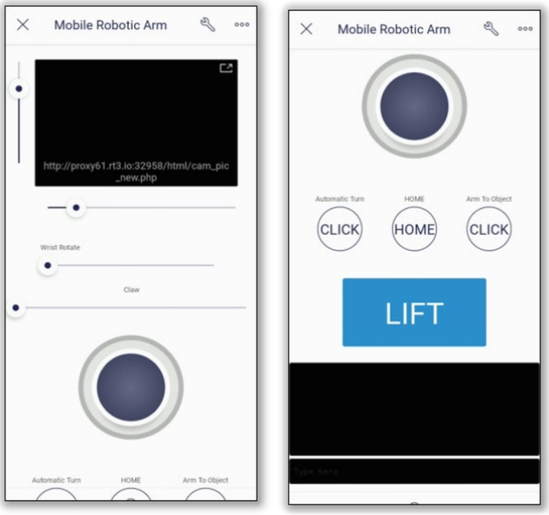

Fig. 9. Mobile App using Blynk IoT platform

The Blynk IoT is a platform providing IoT functionality. This platform also provides a app builder with which we can build our app using its provided widgets.

Featurs of our Mobile Appication created to control the Mobile Arm:

- Camera Output Display
- Horizontal and vertical Sliders to control Arm position
- Sliders To control Wrist Angle and Claw
- Joystick to move the Robot
- Button to perform some programmed operations such as Automatic Turn, Moving Arm to Object Location and Home button to place arm to home location
- LIFT button to lift the arm up
- Terminal window to perform debugging.

III. Working

A. Automatic Arm Movement Towards object

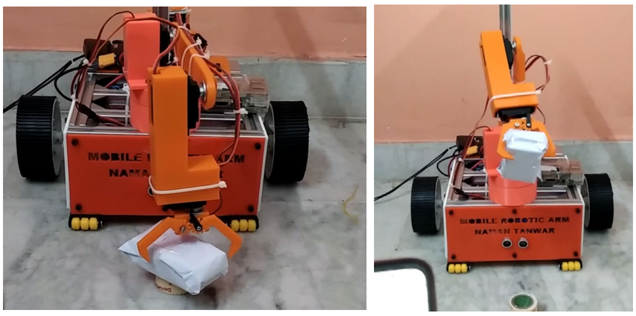

Fig. 10. Automatic Arm Movement and Object Lifting

The front Sensor is used to find the distance of the object from the robot which then is used to move the robotic arm. Then the object is lifted just by clinking on the *LIFT* button on the mobile application.

B. Sensor readings through debug window

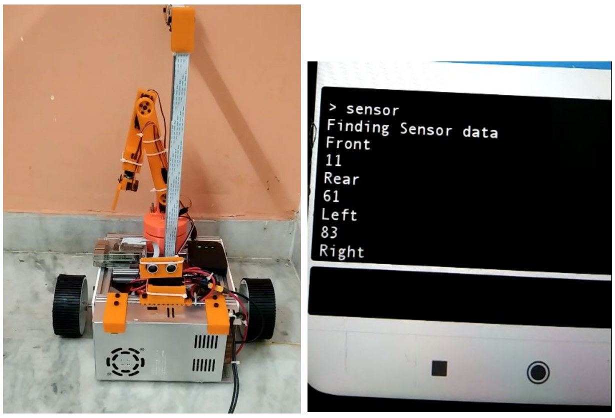

Fig. 11. Robot State and Sensor Readings

The robot is programmed as such if you enter "sensor" in the debug window then the robot will find the distance is all 4 directions and then displays it on the debug window itself.

C. Automatic Turn

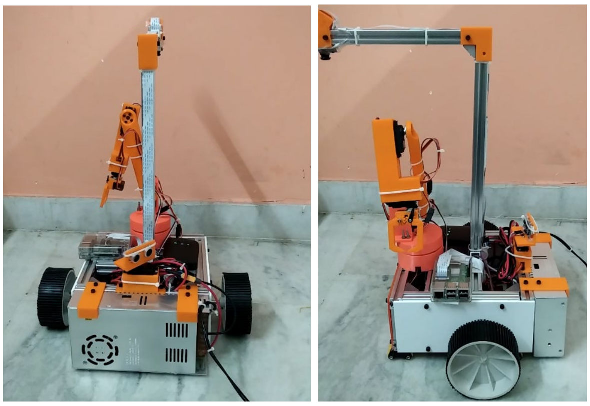

Fig. 12. Robot initial State and State after Turn Operation

The robot will first find the distances in all 4 directions and then turn Right or Left based on the amount of distance available.

The entire robot's operation can be viewed through the demonstration videos. Those Videos and other related stuff can be viewed through the Google Drive link shared. *[6]*

IV. Future Improvements

- Using LIPO battery instead of SMPS Supply
- Using better servos (rather stepper motors) which are more precise, have *zero gear dead-zone* and can handle more weight (This may require a major design change and implementation of gear system)
- Coding an *automatic area map algorithm* to perform certain task in remote area without human intervention
- Using *LIDAR* instead of Ultrasonic sensors to map the area
- Using *image processing* to identify the object with its 3D dimensions

V. References And Useful Links

[1] GitHub link to view this entire project: https://github.com/namanteg/Mobile-Robotic-Arm

[2] GitHuB link for local Raspberry Camera setup: https://github.com/silvanmelchior/RPi_Cam_Web_Interface

[3] Online Laser Cutting service used: https://robu.in/product/online-laser-cutting-service

[4] Blynk IoT platform website: https://blynk.io/

[5] Remote.It website: https://remote.it/

[6] Google Drive link for videos related to robot's operation: https://drive.google.com/drive/folders/1_x-DkT6cmrkyrrUYNz0MS6p55thI3KZy?usp=sharing