A Custom-designed 3D Printed 4-DoF Mobile Robotic Arm for Remote Exploration

Naman Tanwar
School of Electronics Engineering
Vellore Institute of Technology
Vellore, Tamil Nadu, India

Sanjay Kumar Singh
School of Electronics Engineering
Vellore Institute of Technology
Vellore, Tamil Nadu, India

***Abstract*—Exploration of various unreachable/hazardous places is a key to scientifically enriching 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. A custom-designed 3D printed 4DoF Robotic arm is used for operations in the remote terrain with a movable rigid aluminum custom-designed chassis with multiple distance sensors and a camera for remote exploration and navigation.**

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 in 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 for military reasons like spying on terrorist operations.

Hence to serve these purposes, this paper presents a project that aims to build a remotely controlled robot that can explore, navigate, operate, and can be manually controlled in such remote places.

This project is a 3D printed 4-DoF robotic arm mounted on a custom-designed Aluminium-based movable chassis with different sensors and a camera. Ultrasonic sensors are used to provide the available distance in all four directions and a camera is used to visually inspect the target on which the robot will operate. The whole construction can be controlled remotely via the internet. The only requirement is that that place should have mobile internet coverage. This can also affect the places it can operate but different modes of connectivity can be used according to the location where the robot will operate.

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.
- The front *Ultrasonic sensors* will be used to measure the distance of the front obstacle. The distance can then be used by the robotic arm to operate on that obstacle
- The rear *Ultrasonic sensors* are mounted on a *servo motor* which will help to measure the distance from an obstacle in 3 directions using only one sensor
- Wheels and the robot arms are controlled using an *ESP32* microcontroller
- A live camera is set up 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 the *Blynk IoT Platform*

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

B. Mechanical Design and Implementation

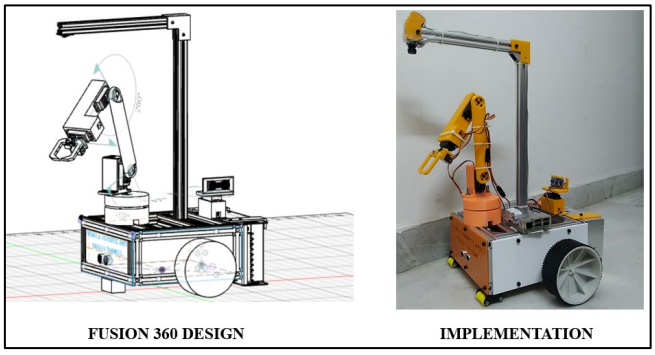

Fig. 1. Mechanical Design and Implementation

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

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

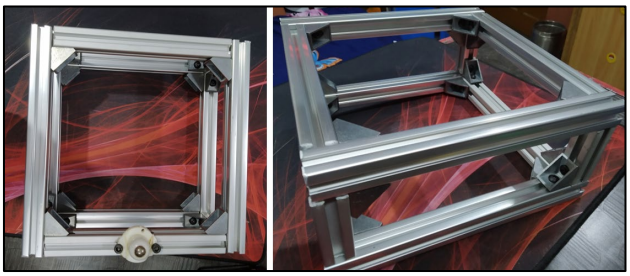

Fig. 2. Robot Chasis

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

The Camera is mounted horizontally to the ground to get a perfectly aligned 2D image of the ground which will be further useful for performing operations on the ground objects via the robotic arm.

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

Two 100rpm geared DC motors mounted on the orange acrylic sheets will be used as main motors to move the entire robot around.

Two wheels are attached to the geared motors to provide mobility. The motors with wheels are mounted horizontally centered concerning the complete chassis (this includes the width of the power supply).

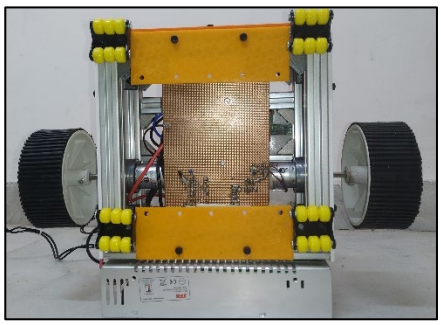

Fig. 3. Robot's bottom view. 4 castor wheels, PCB board, 3D printed PCB holder plates, and 2 main 100mm wheels can be observed.

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:

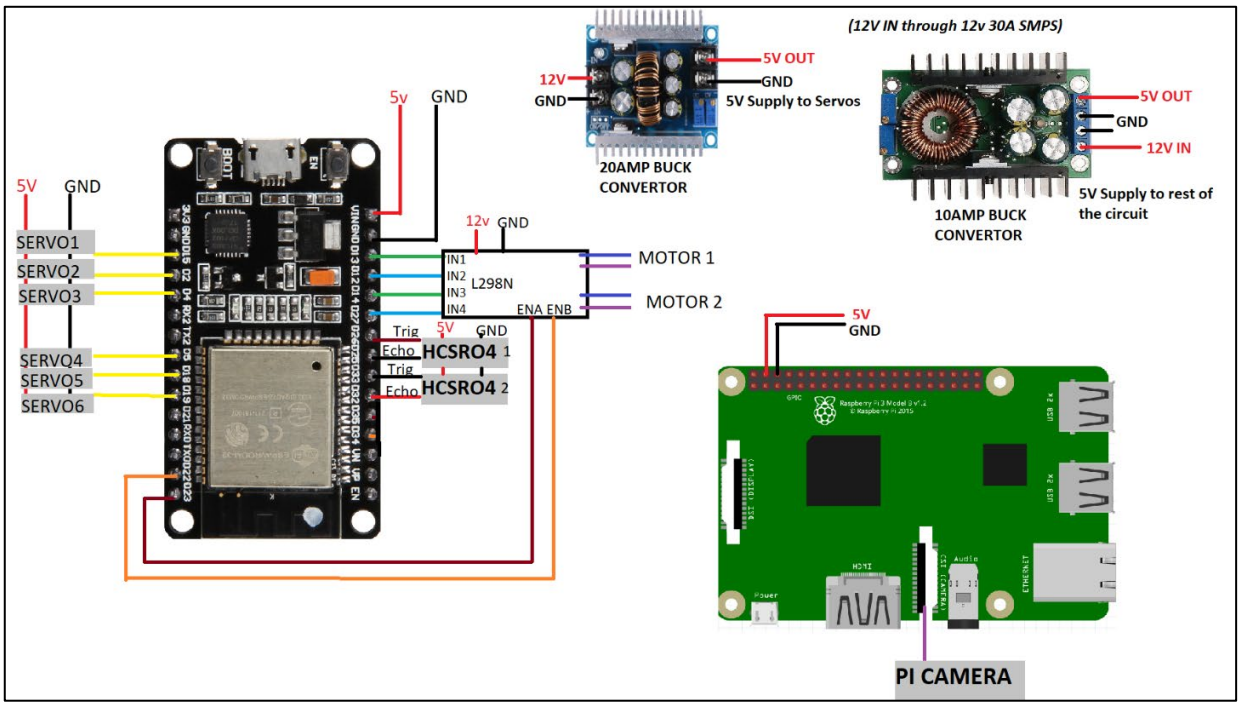

Fig. 4. Circuit Diagram

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. Since this power supply is temporary, It can be replaced with a suitable LiPo battery (preferably, *3S 35C 3500MAh*) for mobility [7].

The voltage is then stepped down to 5v using *Buck Converters* so that it can be used to power other components.

The servo (*Tower Pro MG995*) can pull up to 1.5A to 2A of current under stall conditions. So all the 6 servers can pull up to 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 the rest of the circuit.

ESP32 is used as a microcontroller as it has WiFi functionality which can be used to connect to the internet. A JioFi hotspot is used to connect to the internet. Since the JioFi hotspot is based on the mobile internet, the hotspot itself is mounted on the robot and hence there is no effect on the mobility of the robot

2. ESP32 and connected devices

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

3. Raspberry Pi and Camera

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

The previously used 5v 10Amp buck convertor is used to directly power the raspberry pi through its GPIO pins.

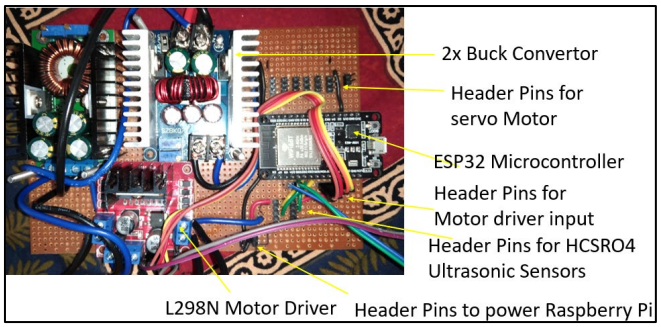

Fig. 5. Circuit

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. *Female headers* are used to mount ESP 32 on the PCB.

Two Buck converters can be seen mounted with some screws on the PCB in the top left. At the bottom, the L298N motor driver is mounted with some screws.

D. Raspberry Pi Camera setup:

- A web interface is set up for the Raspberry Pi camera through an Open Source GitHub Project. But this can only be viewed locally. *[2]*
- Then, using the 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 were 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 closed point for the servo used in the claw was found to be 180 and 140 respectively

F. Movement Equations

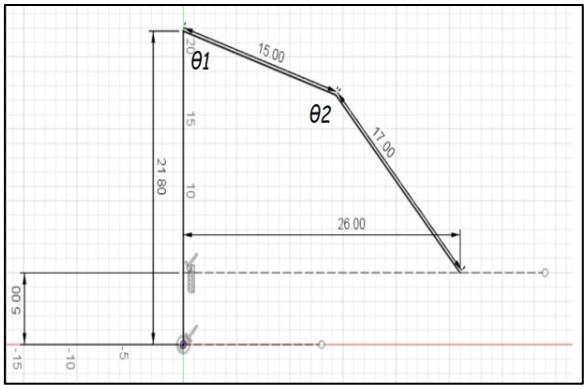

Fig. 6. 2D constrained sketch in Fusion 360

A *2D constrained line schematic* was set up in Fusion 360. For different values of the horizontal distance, angle values were found which then was inserted into Microsoft Excel to find polynomial 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

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

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

G. Mobile App using blynk

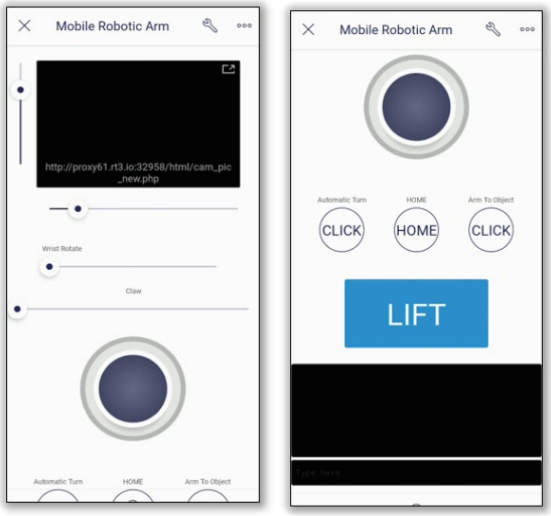

Fig. 8. Mobile App using Blynk IoT platform

The Blynk IoT provides IoT functionality to the robot. This platform also provides an app builder with which we can build our app using its provided widgets.

Features of our Mobile Application:

- Display for camera output
- Horizontal and Vertical Sliders to control Arm position
- Sliders to control Wrist Angle and Claw
- Joystick to move the Robot
- Buttons 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.

H. ESP32 Code Flow chart and Explanation

The code for the ESP32 microcontroller is written in C++ inside Arduino IDE. Different extra libraries for the Blynk platform and servo control were also installed in the Arduino IDE. Since the code has more than 500 lines so only an explanation and flowchart are mentioned in this paper. The actual code can be viewed using the GitHub link provided in the references [1].

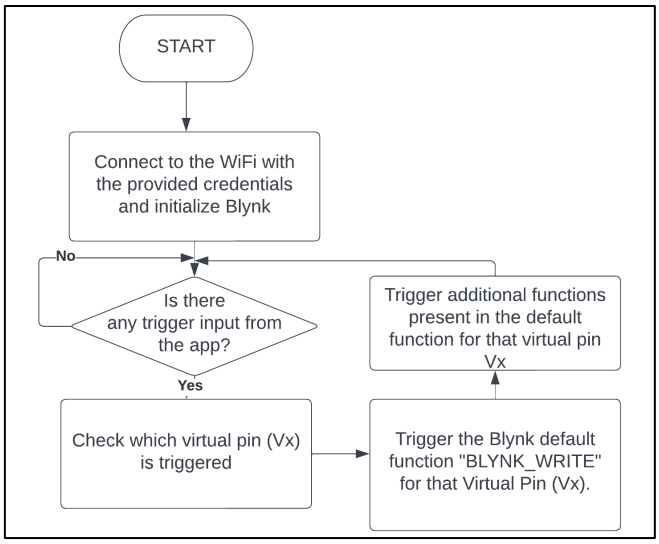

Fig. 9. Basic Flow Chart of the Operation

Initially, the ESP32 gets connected to the WiFi hotspot with the credentials provided in the code itself. Then the device looks for the trigger of any virtual pin through the application created using the Blynk platform.

Vx represents the Virtual Pin where 'x' stands for the pin-number. The various functions involved in the operation are listed in *TABLE I.*

TABLE I. Various Functions Triggered by the Virtual Pins and their Description

Virtual Pin/ Data Stream	**Functional Description of BLYNK_WRITE (Vx)**	**Additional Functions Called during operation**
V0	Slider control for Base i.e., Horizontal Slider	*BaseServoOperate()* – Custom function to rotate base servo
V2	Terminal Window	*GetSensorData()* – To get the data from both ultrasonic sensors
V3	Joystick Control	*FrontDist()* – To get front sensor destance data *forward(), backward(), right(), left(), Brake()* – These functions control the 6 pins of the motor driver which then controls the 2 motors.
V4	Vertical Slider	*calcTheta(vertValue)* – This will calculate theta based on the value of the vertical slider according to the formulae derived earlier.

V5	Slider to Rotate Wrist	*WristServoOperate()* – To operate the wrist servo based on the slider value
V6	Slider to operateClaw	*ClawServoOperate()* - To operate claw servo based on the slider value
V7	Home Button	*Home()* – To make the servo go home positions as defined in the code.
V8	AutoTurn Button	*autoTurn()* – This will initially calculate the distance in all 4 directions and then decide to rotate in the preferred direction.
V9	ArmToObject Button	*FrontDist()* – To obtain a front distance value from the sensor *calcTheta()*- To calculate theta values based on the front sensor and then operate the servos *BaseServoOperate()* – To operate base servo
V10	Lift Button	*ArmServoOperate()* – To rotate the servo based on the two angle values

III. WORKING DEMONSTRATION

A. Automatic Arm Movement Towards object

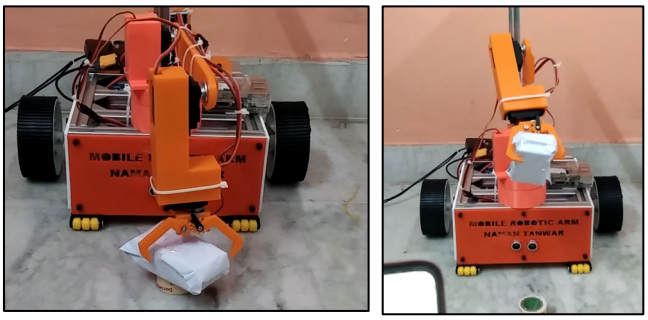

Fig. 10. Automatic Arm Movement and Object Lifting

When the *ArmToObject* button is pressed, the front Sensor finds the distance of the object from the robot which then is used to move the robotic arm. Then the object can be lifted just by clicking on the *LIFT* button on the mobile application.

B. Sensor readings through debug window

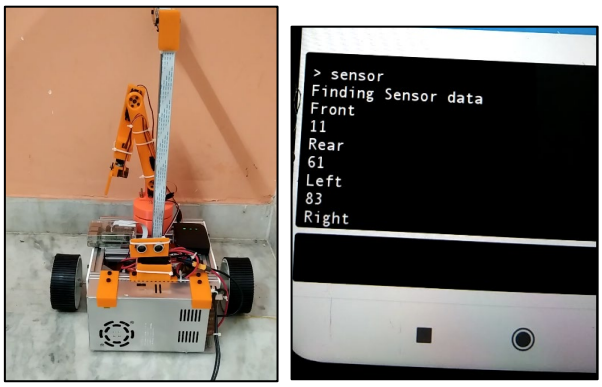

Fig. 11. Robot State and Sensor Readings

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

C. Automatic Turn

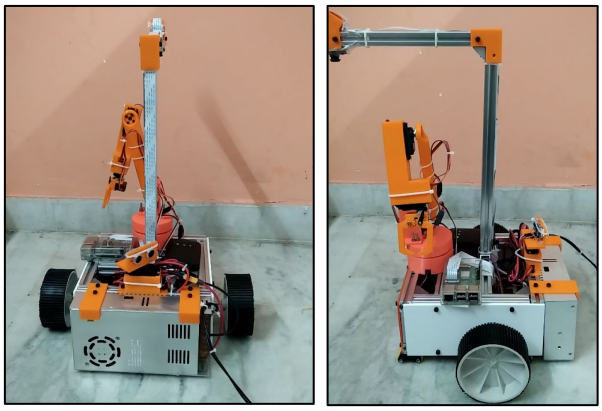

Fig. 12. Robot initial State and State after Turn Operation

When the *Automatic Turn* button is pressed, the robot will first find the distances in all 4 directions and then turn Right or Left based on the amount of distance available.

It was found out that when both the motors are spinning in the same rotational direction (clockwise/anticlockwise) with the same base speed (ie ~100rpm at ~12v), then it takes exactly 600ms to make a perpendicular rotation. Hence to turn left or right the motors were programmed to spin in clockwise or anticlockwise respectively for exactly 600ms.

The entire robot's operation can be viewed through the demonstration videos which can be viewed through the Google Drive link shared. *[6]*

IV. SUMMARY

In this paper, a practical model of the Mobile Robotic Arm was discussed. Initially, the complete robot was designed using Fusion 360 from which some parts were 3D printed including the parts for the Arm of the robot and structural support parts like PCB holder and power supply holder. The design gave an idea of how the real-world robot will look like and it also helped in constructing the Aluminum 2020 profile-based chassis and gave a layout for laser cutting the 3 acrylic

sheets which were then used to mount the ultrasonic sensor and the motors

Then circuit was designed and built on a PCB board (Veroboard) keeping in mind the power requirements of every component. Temporarily a 12v SMPS was used which can easily be replaced with a suitable LiPo battery in the future.

Then the movement equations were practically derived, different moving parts were calibrated and finally, a suitable code for the ESP32 was written in C++ with different functions like Setting arm location using Sliders, Auto Arm to Object, Arm to Home, Lift Operation, Automatic Robot Turn operation, Joystick Movement and Getting data using terminal, using Arduino IDE.

Finally, the working of the complete project was demonstrated.

V. Future Improvements

- Using LIPO battery instead of SMPS Supply
- Using better servos (rather than 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 a certain task in a 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

VI. 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

[7] Battery calculator: https://www.batteriesinaflash.com/c-rating-calculator

[8] Russell Barnes, "Power your Raspberry Pi: expert advice for a supply". Available at: https://magpi.raspberrypi.com/articles/power-supply

[9] Pavel Bayborodin, "How to get started with Blynk", posted on June 15, 2021. Available at: https://blynk.io/blog/how-to-get-started-with-blynk

[10] Official Fusion 360 design guide referred. Available at: https://help.autodesk.com/view/fusion360/ENU/courses/