Project Report on

Manhole Inspection and Debris Removal

Submission to The Robotics Club – SNIST as a part of Post Induction'24

Team No – 11

THE ROBOTICS CLUB – SNIST

SREENIDHI INSTITUTE OF SCIENCE AND TECHNOLOGY

(AUTONOMOUS)

(Affiliated to JNTU University, Hyderabad)

Yamnampet, Ghatkesar, Hyderabad – 501301

2024

CERTIFICATE

This is the project work titled ‘Manhole Inspection and Debris Removal’ by ‘P.Varnan Reddy, G.Rishikesh, G.Varish Teja, N.Selvaraj Das, Devisri Samala, P.Akhila, M.Dharani’, under the execution of ‘B.Sujana’ and is a record of the project work carried out by them during the year 2023-24 as a part of POST INDUCTION’24 under the guidance and super vision of

Mr. B. Navyahas Reddy

&

Mr. P. Phani Anirudh

Joint Secretary of Technical Affairs

Mr. G. Kovidh Addhish

The President of

The Robotics Club

Dr. A. PURUSHOTHAM

Faculty Advisor

ACKNOWLEDGEMENT

This project report is the outcome of the efforts of many people who have driven our passion to explore into the implementation of **MANHOLE INSPECTION AND DEBRIS REMOVAL**. We have received great guidance, encouragement and support from them and have learned a lot because of their willingness to share their knowledge and experience.

We thank our Joint Secretary of Technical Affairs **Mr. B. Navyahas Reddy** and **Mr. P. Phani Anirudh** for being with us till the end of the project completion.

We thank all the members of the **Steering Body, Executive Body, Technical Advisory Board and Club's Incubation and Competence Committee** of **The Robotics Club** for helping us with crucial parts of the project. We are deeply indebted to **Mr. G. Kovidh Addhish** – The President, **Mr. R. Surya Prakash** – The Vice President, **Mr. P. Aarushraj** – General Secretary and **Mr. V. S. S. Sankeerth** – SAB Chairman **THE ROBOTICS CLUB** respectively and also every other person who spared their valuable time without any hesitation whenever we wanted.

We also thank our faculty advisor **Dr. A. Purushotham** Professor Mechanical Department who encouraged us during this project be rendering his help when needed.

DECLARATION

The project work reported in the present thesis titled **“Manhole Inspection and Debris Removal”** is a record of work done by Team_11 in **THE ROBOTICS CLUB** as a part of **POST INDUCTION’24.**

<u>No part of the thesis is copied from books/journals/Internet and wherever the portion is taken, the same has been duly referred in the text. The report is based on the project work done entirely by Team 11 and not copied from any other source.</u>

THE ROBOTICS CLUB-SNIST

TEAM –11

POST INDUCTION’24

MAN HOLE INSPECTION AND DEBRIS REMOVAL

ABSTRACT

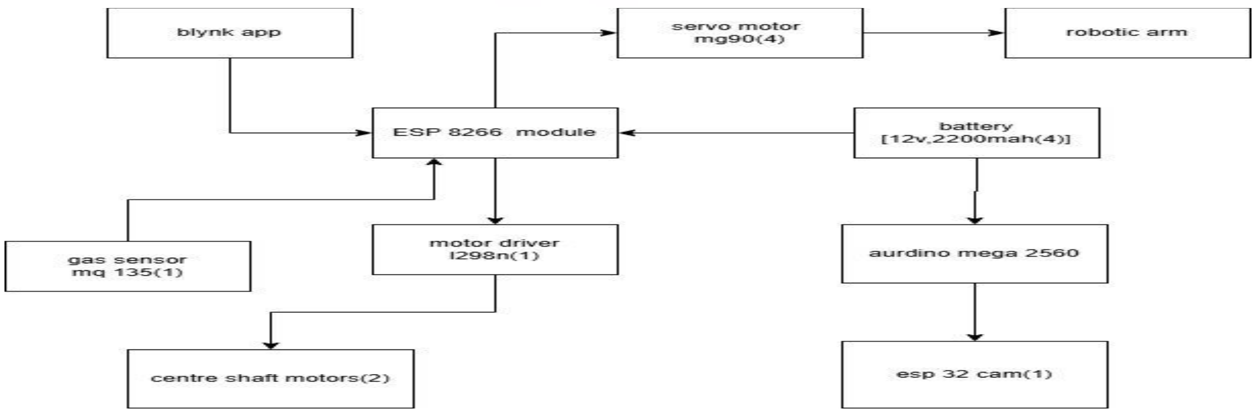
THE PROBLEM:

The present situation poses an extreme health threat, driven by the unhygienic surroundings which is a possibility of disease transmission. Traditional cleaning methods, reliant on human labour, are not exhausting but also riddled with mistakes and inefficiencies. This manual approach is inadequate for ensuring through the cleaning and inspecting man holes. Consequently there is an urgent and undeniable necessity to innovate and implement an automated system. Such a system ensures effective cleaning the path in the man holes without manual intervention. Major problems faced by manholes clogging and blockages, flooding, accidental falls, toxic gases, obstruction to roads, infrequent inspection.

TEAM’S APPROACH TO SOLVE THE PROBLEM:

Addressing these above challenges requires proactive maintenance, advanced monitoring technologies. Considering the above problem, our team has come up with a solution of creating “MAN HOLE INSPECTION AND DEBRIS REMOVAL” a man hole cleaning robot. This robot should be able to move efficiently move through the man holes and carry out the debris in the path effectively. We can use gas sensors for chemical detections. In case of microcontroller, we can use arduino mega. We can equip the bot with wheels for mobility, use motor driver and algorithms for picking up the waste. We can also create feedback mechanisms to adjust the speed or direction to the obstacles.

BLOCK DIAGRAM:



LIST OF CONTENTS

MANHOLE INSPECTION AND DEBRIS REMOVAL

G.Varish Teja, Email: 233110d4@ece.sreenidhi.edu.in

P.Varnan Reddy, Email: 23311a04c8@ece.sreenidhi.edu.in

G.Rishikesh, Email: 23311a04e5@ece.sreenidhi.edu.in

N.Selvaraj Das, Email: 23311a0443@ece.sreenidhi.edu.in

Devisri Samala, Email: 23311a0444@ece.sreenidhi.edu.in

G.Akhila, Email: 23311a0417@ece.sreenidhi.edu.in

M.Dharani, Email: 23311a0447@ece.sreenidhi.edu.in

ABSTRACT:

The present situation poses an extreme health threat, driven by the unhygienic surroundings which is a possibility of disease transmission. Traditional cleaning methods, reliant on human labour, are not exhausting but also riddled with mistakes and inefficiencies. This manual approach is inadequate for ensuring through the cleaning and inspecting man holes. Consequently there is an urgent and undeniable necessity to innovate and implement an automated system. Such a system ensures effective cleaning the path in the man holes without manual intervention. Major problems faced by man holes clogging and blockages, flooding, accidental falls, toxic gases, obstruction to roads, infrequent inspection.

1.INTRODUCTION:

In present days, maintaining manhole has become a crucial part. Manholes containing hazardous gases in it and containing various types of problems in it, effecting the functionality and overall efficiency and there are some of the major problems faced in the manholes are clogging, blockages, Infiltration and Exfiltration, flooding, Accidental falls, Toxic gases, Obstruction to roads. Traditional cleaning methods involve manual labour, health risks to the workers who are often exposed to the hazardous conditions. Beyond the health implications, these conventional methods are time-consuming, labour intensive, and often lack the precision and thoroughness required for better maintenance.

To address these challenges, we are developing a Manhole Inspection and Debris Removal robot designed to revolutionize the above mentioned problems. Combining the capabilities of robotics, real time video streaming and sensor technology. Operator will be getting the real time feedback while it is operating inside the manhole

The project aims not only to enhance the efficiency and effectiveness of inspection and debris removal but also to prioritize the safety of personnel by minimizing direct human intervention in that environment. In which some of them can be rectified by using the robotic arm.

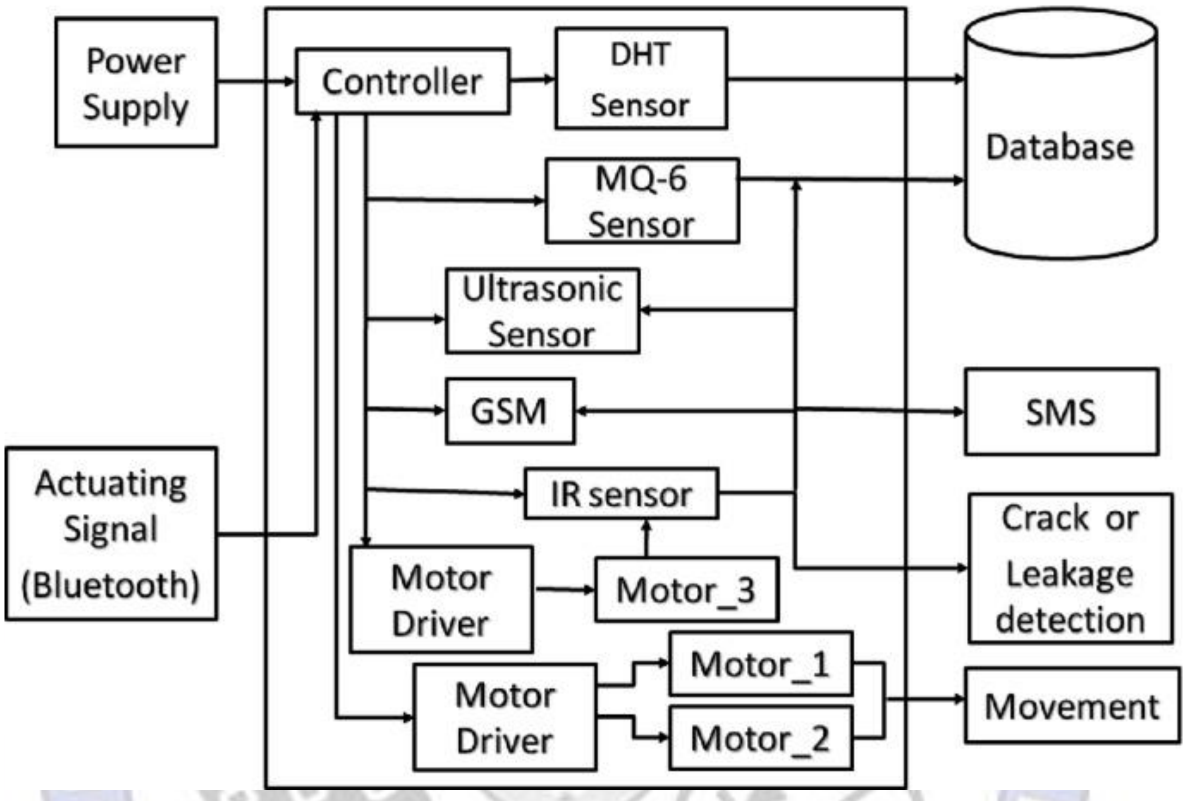
1.PURPOSE OF THE PROJECT

The main purpose of the project is focused on Manhole inspection and Debris Removal is to enhance the efficiency, effectiveness and maintaining and monitoring the manhole systems. Removing the debris causing clogging, blockage of the way using the robotic arm and mainly to know the harmful gases present inside the manhole, and to know the possibility of human intervention.

1.1 EXISTING SYSTEMS

A prototype of a pipeline inspection robot is proposed in this paper which can go inside a specific set of pipe sizes. This mobile robot can go horizontally and bend up to 30° inclined pipelines and is controlled by either a PC or a Smart Phone. It is well equipped with the image capturing capability to provide images of the faults inside the pipeline and for further processing. It has the capability to measure the barrier or the amount of scale inside the pipeline using its ultrasonic sensor. Figure 1 illustrates the block diagram of the proposed system

1.3 BLOCK DIAGRAM OF EXISTING SYSTEMS



1.4 LIMITATIONS OF EXISTING SYSTEMS

1. The robot can only operate within a specific set of pipe sizes. This limits its applicability for inspecting pipelines.

2. The robot can only traverse pipes that are inclined up to 30°. Pipelines with steeper inclines may not be accessible to the robot, limiting its effectiveness in certain pipeline systems with varying angles.

3. The robot can be controlled via a PC or a smartphone, which implies reliance on the robustness and reliability of wireless communication. Interference, connection loss, or range issues may impact the robot's control, especially in large or complex pipeline networks.

4. While the robot has imaging and ultrasonic sensors to detect faults and measure scale, that require more advanced sensors or technology for accurate detection.

5. The robot's ability to traverse pipelines and perform inspections may be constrained by the available power source, especially in long pipelines

2. LITERATURE REVIEW

2.1 LITERATURE SURVEY

Recent literature emphasizes advancements in Manhole Inspection and debris Removal. Integral tasks for maintaining the functionality of, particularly debris containing water systems. These systems require regular maintenance to prevent blockages, leaks, and structural failures that could lead to significant public health and environmental risks. Traditionally, these tasks have been performed manually, but recent advancements in robotics and automation have revolutionized the process. This literature review examines key research and technological advancements in robotic systems for manhole inspection and debris removal.

3. PROPOSED SYSTEM

3.1 PROPOSED SYSTEM

The robot utilizes a ESP8266 microcontroller. The robotic arm for the arm movement inside the manhole, the motor drivers used are L298N for the robot movement and MQ135 gas sensors to know the hazardous gases present inside the manhole can be recognized by the help of this gas sensor and these are connected to the microcontroller which is linked using the BLYNK app where operator can know the required feedback regarding the bot. The robot is fitted with geared center shaft motors. Robot with the capability to extend arm doesn't damage itself.one of the main features of our robot is its real-time video monitoring capability, facilitated through the ESP32-CAM which is connected to the ARDUINO MEGA and can visualize through the web server. Further enhancing the user experience and control, we integrated the robot with the mobile applications. Where it offers platform for both monitoring the video and remotely controlling the robot's movements and functions.

3.2 ADVANTAGES OF PROPOSED SYSTEMS

The advantages of the Proposed Systems is, by the help of modern technology used in the robot, operator can know the feedback of the internal system of the manhole. The harmful gases inside the manhole pipelines can be deducted which is hazardous for the human intervention. Mainly the robotic arm which removes the debris inside the pipelines. The operator will be having the live video streaming of the robot inside the manhole. These are some of the advantages of the proposed system

3.3 HARDWARE REQUIREMENTS

3.3.1 Node MCU ESP8266

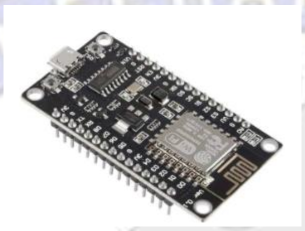


Figure 1: NodeMCU ESP8266

NodeMCU ESP8266 is an open-source, low-cost Wi-Fi module with integrated microcontroller capabilities. It's based on the ESP8266 and is widely used in IoT (Internet of Things) projects. NodeMCU makes it easy to connect and program IoT devices, offering a simple and accessible platform for building Wi-Fi-enabled projects. It supports the Arduino IDE, making it popular among developers and hobbyists for creating connected applications and prototypes.

3.3.2 BREAD BOARD

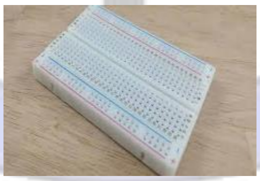


Figure 2: Bread Board

An electronics breadboard (as opposed to the type on which sandwiches are made) is actually referring to a solder-less breadboard. These are great units for making temporary circuits and prototyping, and they require absolutely no soldering.

3.3.3 MQ135 GAS SENSOR

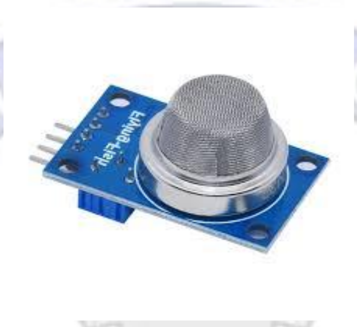


Figure 3: MQ135-Gas Sensor

The MQ135 gas sensor is a popular device used for detecting various gases in the air, including Ammonia (NH_3), Sulphur, Benzene, Smoke, Alcohol, and Carbon dioxide (CO_2). The MQ-135 gas sensor operates on the principle of changes in resistance when it comes into contact with the target gas. As the concentration of the gas increases, the sensor’s resistance decreases

3.3.4 CENTER SHAFT MOTOR

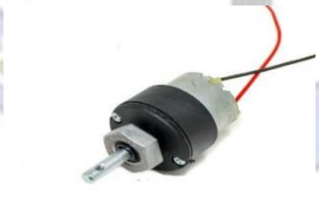


Figure 4: Center shaft motor

The single Shaft Motor - is an electro-mechanical device that converts electrical energy from a direct current power source into mechanical motion.They are known for their simplicity, reliability, and ease of control, making them suitable for various tasks.

3.3.5 ESP32-CAM

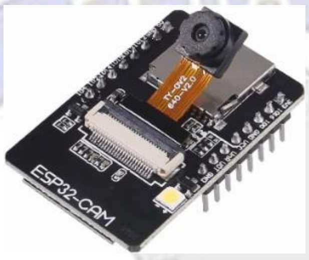


Figure 5: ESP32-CAM

The ESP32-CAM is a versatile development board that combines the ESP32 microcontroller with a camera module. It comes with an OV2640 camera. The ESP32-CAM is widely used for projects involving video streaming, image capture, and home security applications, making it a valuable tool in the realm of embedded systems and the Internet of Things (IOT).

3.3.6 MOTOR DRIVER – L298N

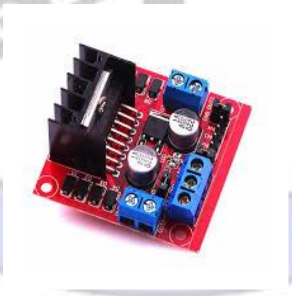


Figure 6: L298N Motor Driver

The L298N is a dual H-bridge motor driver integrated circuit (IC) commonly used to control and drive DC motors, particularly in robotics and mechanics applications. It provides a straightforward way to manage motor direction and speed using control signals from micro-controllers like Arduino. The L298N motor driver has a supply range of 5v to 35v and is capable of 2A continuous current per channel, so it works well with most of the DC motors. The L298N is valued for its efficiency, ease of use, and versatility, making it a popular choice for hobbyists and engineers working on projects involving motor control.

3.3.7 LI-ION BATTERIES

Figure 7: Li-Ion Batteries

A lithium-ion or Li-ion battery is a type of rechargeable battery which uses the reversible reduction of lithium ions to store energy. The anode (negative electrode) of a conventional lithium-ion cell is typically graphite made from carbon. The cathode (positive electrode) is typically a metal oxide. The electrolyte is typically a lithium salt in an organic solvent. It is one of the rechargeable batteries. It exhibits lower self-discharge rate.

3.3.8 SERVO MOTORS MG90S

Figure 8: Servo Motor MG90

It is a micro servo motor that operates on precise angular positioning. It converts electrical signals into mechanical motion with high accuracy. The speed of the motor reduces and torque increases. The DC motor provides the rotational force. This will be operating by the operator for the correct bot movement. And will be getting the correct feedback.

3.3.9 JUMPER WIRES

Figure 9: Jumper Wires

A jumper wire is an electrical wire or group of wires used to connect circuits without soldering. They have connectors or pins at their ends. Depending upon the configuration of end connectors, they are classified into three types: maleto-male, male-to-female and female-to-female.

3.3.10 WHEELS

Figure 10: Wheels

Wheels with good grip are used so that the bot moves freely through the pipes. They reduce friction, making it easier to transport heavy loads and navigate obstacles. We can traverse through rough paths as they have good grip. The ones we are using are made of plastic and rubber.

3.3.11 CELL HOLDER

Figure 11: Cell Holder

It is a device designed to hold the batteries or cells, ensuring that they can deliver power to a circuit or device. It acts as an interface between the battery and the electronic system. It is connected to the correct polarity.

3.3.12 LED LIGHTS

Figure 12: LED Lights

3.3.13 ARDUINO MEGA

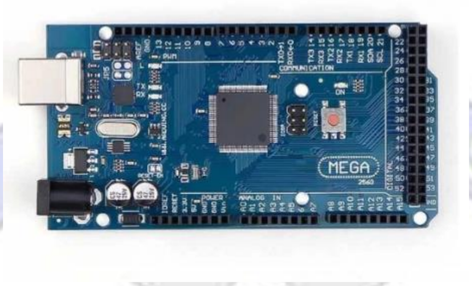


Figure 13: Arduino Mega

The Arduino Mega 2560 is a microcontroller board based on ATmega2560.It is designed for more complex projects and is ideal for application requiring a large number of input or output pins and memory. It’s operating voltage is 5V and input voltage is which is recommended is 7 -12V. It’s input voltage limit is 6-20V, it contains 54 I/O pins out of which 15 can provide PWM pins and 16 are Analog input pins. It’s Flash Memory is 256KB. It contains a SRAM of 8KB and EEPROM of 4KB. And contains a clock speed of 16MHz.

3.4 SOFTWARE REQUIREMENTS

3.4.1 ARDUINO IDE

sketch_mar26a | Arduino 1.8.12
sketch_mar26a

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Arduino Portenta H7 (M7 core) on /dev/cu.usbmodem141101

Figure 14: Arduino IDE

Arduino Integrated Development Environment is an opensource application software created by Arduino. It is used to write and upload code on to the Arduino boards. It supports C and C++ programming languages, and has a built-in compiler. You can compile your code within the IDE to check for errors and then upload the compiled program (known as a sketch) to the connected Arduino board. The IDE handles the compilation and uploading process easily.

3.4.2 FUSION 360

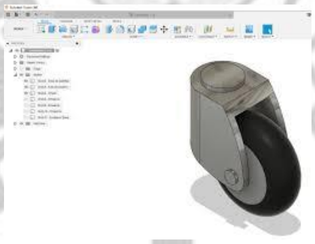


Figure 15: Fusion 360

Fusion 360 is a computer-aided designing (CAD) software application for 3-D modelling and simulation. Its other functions include computer-aided manufacturing (CAM) and computer-aided engineering (CAE), as well as designing printed circuit boards. Fusion 360 provides powerful 3D modelling tools that allow users to create complex 3D models of products and components. It supports parametric modelling, direct modelling, and sculpting.

3.4.3 FRITZING

Figure 16: Fritzing

Fritzing is an open-source electronic design automation (EDA) software to design electronics hardware, printed circuit boards and schematic circuit diagrams. It is an offshoot of the Processing programming language and the Arduino microcontroller. Fritzing allows users to create electronic circuits by dragging and dropping various components (e.g., resistors, LEDs, microcontrollers) onto a virtual breadboard. Users can connect components by drawing wires to represent electrical connections.

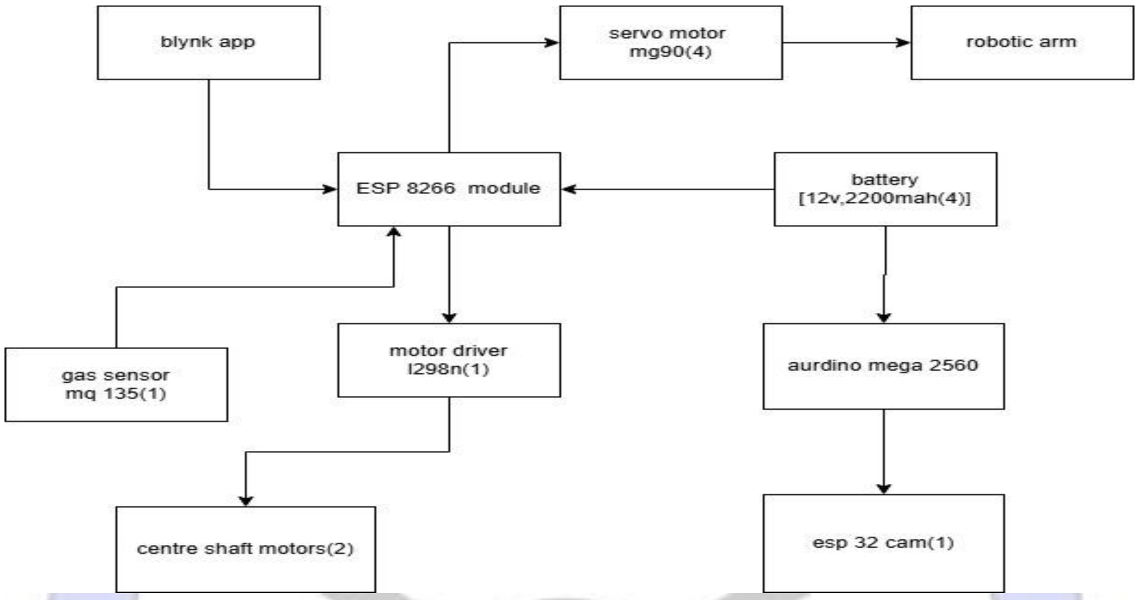
3.4.5 BLYNK IOT

Figure 17: Blynk IOT

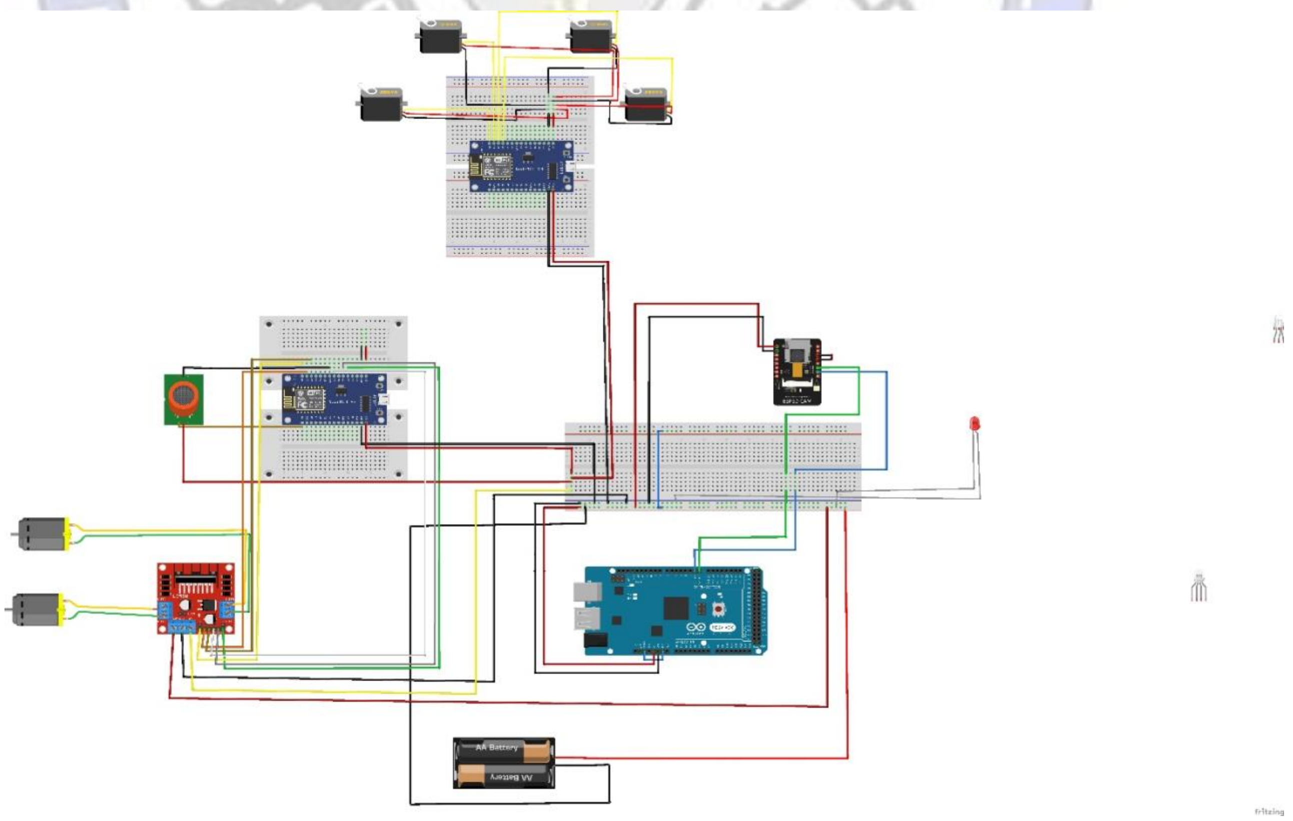
Blynk is an IOT platform for iOS or Android smartphones that is used to control micro-controllers. This application is used to create a graphical interface or human machine interface by compiling and providing the appropriate address.We will be able to get the appropriate feedback using this app.

4.WORKING OF PROPOSED SYSTEM

4.1 BLOCK DIAGRAM



4.2 CIRCUIT DIAGRAM



4.3 WORKING MECHANISM

The Manhole Inspection robot is designed to navigate through manhole pipelines, providing real-time video feedback and performing cleaning operations while monitoring toxic gas levels. Upon activation, the robot begins by establishing a Wi-Fi connection, allowing the bot movement, robotic arm movement and MQ135 gas sensor which detects the harmful gas which is connected using the BLYNK app where operator can monitor the feedback in the app. For the video streaming we are using a ESP32-CAM module which is connected to the ARDUINO MEGA for the video-streaming, offering operator a clear view of the manhole environment. As the robot moves, the two geared DC motors, which drive the robot, receive commands from the user interface. Depending on the command, the robot can move forward, backward, left, or right. The robotic arm of the robot will be used to remove the debris inside the manhole. The MQ135 gas sensor continuously monitors the harmful gas levels in the manhole. If the levels rise above a certain threshold, an alert can be sent to the operator, signalling a potential hazard. The entire operation is seen through a user interface, allowing operators to control the robot, view live video feeds, receive harmful gas alerts, and make informed decisions in real-time.

5.RESULTS

5.1 RESULTS

Upon the completion and deployment of the Manhole Inspection and Debris Removal, remarkable improvements in manhole maintenance were observed. One of the primary achievement of the project was elimination of direct human interventions. Due to which toxic gas exposure, infections and threats to the health were reduced. As the bot is equipped with the robotic arm which is used to remove the debris near the clogging or blockages etc. Due to the video streaming we were even able to inspect the manhole pipe inside to see whether if there is any damage. The gas sensor allowed us to monitor the gas levels and take the actions accordingly. Overall, the bot was a success as it was a cost effective and easy to use and understand the controls.

6.CONCLUSION AND FUTURE SCOPE

6.1 CONCLUSION

The Manhole Inspection and Debris Removal aims to know the condition inside the manhole pipelines in a efficiency and effective manner. Traditional methods have limitations, including manual labour in the hazardous conditions. This robot, using the modern technology, semi-autonomous and gives the solution accordingly. And is also for inspecting and removing the unwanted debris.

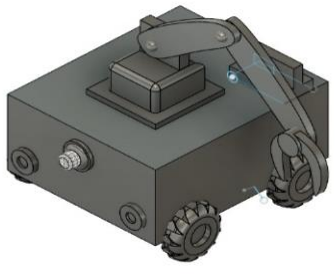


Figure: CAD Design of prototype

6.2 FUTURE SCOPE

The future of pipeline inspection and debris removal using robotic arms holds significant promise with advancements in several key areas. Robots are expected to evolve with enhanced autonomous navigation, utilizing AI and machine learning to adapt to complex pipeline networks with minimal human intervention. Integration of multi-sensor technology, including infrared, thermal, and 3D imaging, will enable more precise detection of pipeline issues, such as cracks, corrosion, and leaks. The robotic arms could also improve debris removal with advanced tools like high-pressure jets or chemical treatments for more efficient cleaning. Real-time data processing and AI will enable instant decision-making, improving maintenance schedules and predicting failures before they occur. Future systems may also involve collaborative swarm robotics for large-scale inspections and digital twins for predictive maintenance. Enhanced durability, remote operation capabilities, and improved safety features will make these robots suitable for harsher environments, significantly reducing human risk while optimizing pipeline maintenance.

SOURCE CODE:

```
#define BLYNK_TEMPLATE_ID "TMPL3I6vFNGrg"
#define BLYNK_TEMPLATE_NAME "Quickstart Template"
#define BLYNK_AUTH_TOKEN "aAlnQc01vlv38yoqpcCYpPSAuZ4zO1kn"

#include <BlynkSimpleEsp8266.h>
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <Servo.h>

// Wi-Fi credentials
char ssid[] = "realme 11 Pro 5G";
char pass[] = "12345678@";


//
// WARNING!!! PSRAM IC required for UXGA resolution and high JPEG quality
//            Ensure ESP32 Wrover Module or other board with PSRAM is selected
//          Partial images will be transmitted if image exceeds buffer size
//
//          You must select partition scheme from the board menu that has at least 3MB APP
space.
//          Face Recognition is DISABLED for ESP32 and ESP32-S2, because it takes up from
15
//          seconds to process single frame. Face Detection is ENABLED if PSRAM is enabled
as well


// ===================
// Select camera model
// ===================
//#define CAMERA_MODEL_WROVER_KIT // Has PSRAM
//#define CAMERA_MODEL_ESP_EYE  // Has PSRAM
//#define CAMERA_MODEL_ESP32S3_EYE // Has PSRAM
//#define CAMERA_MODEL_M5STACK_PSRAM // Has PSRAM
//#define CAMERA_MODEL_M5STACK_V2_PSRAM // M5Camera version B Has
PSRAM
//#define CAMERA_MODEL_M5STACK_WIDE // Has PSRAM
//#define CAMERA_MODEL_M5STACK_ESP32CAM // No PSRAM
//#define CAMERA_MODEL_M5STACK_UNITCAM // No PSRAM
//#define CAMERA_MODEL_M5STACK_CAMS3_UNIT  // Has PSRAM
#define CAMERA_MODEL_AI_THINKER // Has PSRAM
//#define CAMERA_MODEL_TTGO_T_JOURNAL // No PSRAM
//#define CAMERA_MODEL_XIAO_ESP32S3 // Has PSRAM
// ** Espressif Internal Boards **
//#define CAMERA_MODEL_ESP32_CAM_BOARD
//#define CAMERA_MODEL_ESP32S2_CAM_BOARD
//#define CAMERA_MODEL_ESP32S3_CAM_LCD
//#define CAMERA_MODEL_DFRobot_FireBeetle2_ESP32S3 // Has PSRAM
//#define CAMERA_MODEL_DFRobot_Romeo_ESP32S3 // Has PSRAM
#include "camera_pins.h"
```

```
// ===========================
// Enter your WiFi credentials
// ===========================
const char *ssid = "realme 11 Pro 5G";
const char *password = "12345678@";

void startCameraServer();
void setupLedFlash(int pin);
// Motor control pins
const int motor1A = 21; // IN1
const int motor1B = 19; // IN2
const int enA = 26;     // ENA
const int motor2A = 22; // IN3
const int motor2B = 23; // IN4
const int enB = 32;     // ENB

// Define Servo objects
Servo servo1;  // Base Servo
Servo servo2;  // Elbow Servo
Servo servo3;  // Wrist Servo
Servo servo4;  // Gripper Servo

// Virtual Pins for Blynk sliders
#define VIRTUAL_PIN_0 V4  // For Servo 1 (Base)
#define VIRTUAL_PIN_1 V5  // For Servo 2 (Elbow)
#define VIRTUAL_PIN_2 V6  // For Servo 3 (Wrist)
#define VIRTUAL_PIN_3 V7  // For Servo 4 (Gripper)

// Input Pins for servos (as per your requirement)
```

```
#define SERVO_PIN_1 D0  // Base servo connected to D0

#define SERVO_PIN_2 D1  // Elbow servo connected to D1

#define SERVO_PIN_3 D3  // Wrist servo connected to D3

#define SERVO_PIN_4 D4  // Gripper servo connected to D4


// Default angles (starting position of servos)

int base_angle = 0;   // Base servo: 0 to 360 degrees

int elbow_angle = 0;  // Elbow servo: 0 to 90 degrees

int wrist_angle = 0;  // Wrist servo: 0 to 45 degrees

int gripper_angle = 0; // Gripper servo: 0 to 90 degrees


// Gas sensor pin and virtual pin

const int mq135Pin = A0; // MQ-135 connected to analog pin A0

#define GAS_SENSOR_VIRTUAL_PIN V8


// Function declarations

void forward();

void backward();

void left();

void right();

void stopMotorsDrive();


void setup() {

  Serial.begin(115200); // Initialize serial communication

// Start serial communication for debugging

  Serial.begin(115200);


  // Connect to Wi-Fi

  WiFi.begin(ssid, pass);

  while (WiFi.status() != WL_CONNECTED) {
```

```
  delay(1000);
  Serial.print(".");
}
Serial.println("Connected to Wi-Fi");

// Connect to Blynk
Blynk.begin(auth, ssid, pass);
Serial.println("Blynk Connected");

// Attach servos to their respective pins
servo1.attach(SERVO_PIN_1);  // Base servo
servo2.attach(SERVO_PIN_2);  // Elbow servo
servo3.attach(SERVO_PIN_3);  // Wrist servo
servo4.attach(SERVO_PIN_4);  // Gripper servo

// Set initial servo positions
servo1.write(base_angle);
servo2.write(elbow_angle);
servo3.write(wrist_angle);
servo4.write(gripper_angle);

// Connect to Wi-Fi
WiFi.begin(ssid, pass);
while (WiFi.status() != WL_CONNECTED) {
  delay(1000);
  Serial.print(".");
}
Serial.println("\nConnected to Wi-Fi");

// Connect to Blynk
```

```
  Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);
  Serial.println("Blynk Connected");

  // Motor control pin setup
  pinMode(motor1A, OUTPUT);
  pinMode(motor1B, OUTPUT);
  pinMode(motor2A, OUTPUT);
  pinMode(motor2B, OUTPUT);
  pinMode(enA, OUTPUT);
  pinMode(enB, OUTPUT);

  // Gas sensor pin setup
  pinMode(mq135Pin, INPUT);
}

BLYNK_WRITE(V1) { // Forward Button
  int value = param.asInt();
  if (value == 1) {
    Serial.println("Forward ON");
    forward();
  } else {
    Serial.println("Forward OFF");
    stopMotorsDrive();
  }
}

BLYNK_WRITE(V2) { // Backward Button
  int value = param.asInt();
  if (value == 1) {
    Serial.println("Backward ON");
    backward();
  } else {
    Serial.println("Backward OFF");
    stopMotorsDrive();
  }
}

BLYNK_WRITE(V3) { // Left Button
  int value = param.asInt();
  if (value == 1) {
    Serial.println("Left ON");
    left();
  } else {
    Serial.println("Left OFF");
    stopMotorsDrive();
  }
}

BLYNK_WRITE(V4) { // Right Button
  int value = param.asInt();
  if (value == 1) {
    Serial.println("Right ON");
    right();
  } else {
    Serial.println("Right OFF");
    stopMotorsDrive();
  }

Serial.setDebugOutput(true);
  Serial.println();
```

```
camera_config_t config;
config.ledc_channel = LEDC_CHANNEL_0;
config.ledc_timer = LEDC_TIMER_0;
config.pin_d0 = Y2_GPIO_NUM;
config.pin_d1 = Y3_GPIO_NUM;
config.pin_d2 = Y4_GPIO_NUM;
config.pin_d3 = Y5_GPIO_NUM;
config.pin_d4 = Y6_GPIO_NUM;
config.pin_d5 = Y7_GPIO_NUM;
config.pin_d6 = Y8_GPIO_NUM;
config.pin_d7 = Y9_GPIO_NUM;
config.pin_xclk = XCLK_GPIO_NUM;
config.pin_pclk = PCLK_GPIO_NUM;
config.pin_vsync = VSYNC_GPIO_NUM;
config.pin_href = HREF_GPIO_NUM;
config.pin_sccb_sda = SIOD_GPIO_NUM;
config.pin_sccb_scl = SIOC_GPIO_NUM;
config.pin_pwdn = PWDN_GPIO_NUM;
config.pin_reset = RESET_GPIO_NUM;
config.xclk_freq_hz = 20000000;
config.frame_size = FRAMESIZE_UXGA;
config.pixel_format = PIXFORMAT_JPEG;  // for streaming
//config.pixel_format = PIXFORMAT_RGB565; // for face detection/recognition
config.grab_mode = CAMERA_GRAB_WHEN_EMPTY;
config.fb_location = CAMERA_FB_IN_PSRAM;
config.jpeg_quality = 12;
config.fb_count = 1;

// if PSRAM IC present, init with UXGA resolution and higher JPEG quality
```

```
  //                      for larger pre-allocated frame buffer.
  if (config.pixel_format == PIXFORMAT_JPEG) {
    if (psramFound()) {
      config.jpeg_quality = 10;
      config.fb_count = 2;
      config.grab_mode = CAMERA_GRAB_LATEST;
    } else {
      // Limit the frame size when PSRAM is not available
      config.frame_size = FRAMESIZE_SVGA;
      config.fb_location = CAMERA_FB_IN_DRAM;
    }
  } else {
    // Best option for face detection/recognition
    config.frame_size = FRAMESIZE_240X240;
#if CONFIG_IDF_TARGET_ESP32S3
    config.fb_count = 2;
#endif
  }

#if defined(CAMERA_MODEL_ESP_EYE)
  pinMode(13, INPUT_PULLUP);
  pinMode(14, INPUT_PULLUP);
#endif

  // camera init
  esp_err_t err = esp_camera_init(&config);
  if (err != ESP_OK) {
    Serial.printf("Camera init failed with error 0x%x", err);
    return;
  }
  sensor_t *s = esp_camera_sensor_get();
  // initial sensors are flipped vertically and colors are a bit saturated
  if (s->id.PID == OV3660_PID) {
    s->set_vflip(s, 1);        // flip it back
    s->set_brightness(s, 1);   // up the brightness just a bit
    s->set_saturation(s, -2);  // lower the saturation
  }
  // drop down frame size for higher initial frame rate
  if (config.pixel_format == PIXFORMAT_JPEG) {
    s->set_framesize(s, FRAMESIZE_QVGA);
  }

#if defined(CAMERA_MODEL_M5STACK_WIDE) ||
defined(CAMERA_MODEL_M5STACK_ESP32CAM)
  s->set_vflip(s, 1);
  s->set_hmirror(s, 1);
#endif

#if defined(CAMERA_MODEL_ESP32S3_EYE)
  s->set_vflip(s, 1);
#endif

// Setup LED FLash if LED pin is defined in camera_pins.h
#if defined(LED_GPIO_NUM)
  setupLedFlash(LED_GPIO_NUM);
#endif

  WiFi.begin(ssid, password);
  WiFi.setSleep(false);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");

  startCameraServer();

  Serial.print("Camera Ready! Use 'http://");
  Serial.print(WiFi.localIP());
  Serial.println("' to connect");
}

}

void loop() {
  // Maintain Blynk connection
  Blynk.run();
}

  // Read the gas sensor value
  int sensorValue = analogRead(mq135Pin);

  // Send the sensor value to the Blynk app
  Blynk.virtualWrite(GAS_SENSOR_VIRTUAL_PIN, sensorValue);

  // Print the sensor value to the Serial Monitor
  Serial.print("Gas Sensor Value: ");
  Serial.println(sensorValue);
  // Wait for a second before reading again
  delay(1000);
}
// Blynk function to control Servo 1 (Base)
BLYNK_WRITE(VIRTUAL_PIN_0) {
  base_angle = param.asInt();  // Get the value from the slider
  // Ensure base_angle stays between 0 and 360
  if (base_angle > 360) base_angle = 360;
  if (base_angle < 0) base_angle = 0;
  servo1.write(base_angle);    // Move base servo to the specified angle
  Serial.print("Base Angle: ");
  Serial.println(base_angle);
}

// Blynk function to control Servo 2 (Elbow)
BLYNK_WRITE(VIRTUAL_PIN_1) {
  elbow_angle = param.asInt();  // Get the value from the slider
  // Ensure elbow_angle stays between 0 and 90
  if (elbow_angle > 180) elbow_angle = 180;
  if (elbow_angle < 0) elbow_angle = 0;
  servo2.write(elbow_angle);    // Move elbow servo to the specified angle
  Serial.print("Elbow Angle: ");
  Serial.println(elbow_angle);
}

// Blynk function to control Servo 3 (Wrist)
BLYNK_WRITE(VIRTUAL_PIN_2) {
  wrist_angle = param.asInt();  // Get the value from the slider
  // Ensure wrist_angle stays between 0 and 45
  if (wrist_angle > 180) wrist_angle = 180;
  if (wrist_angle < 0) wrist_angle = 0;
  servo3.write(wrist_angle);    // Move wrist servo to the specified angle
  Serial.print("Wrist Angle: ");
  Serial.println(wrist_angle);
}

// Blynk function to control Servo 4 (Gripper)
BLYNK_WRITE(VIRTUAL_PIN_3) {
  gripper_angle = param.asInt();  // Get the value from the slider
  // Ensure gripper_angle stays between 0 and 90
  if (gripper_angle > 180) gripper_angle = 180;
  if (gripper_angle < 0) gripper_angle = 0;
  servo4.write(gripper_angle);    // Move gripper servo to the specified angle
  Serial.print("Gripper Angle: ");
  Serial.println(gripper_angle);
}

// Motor control functions
void forward() {
  analogWrite(enA, 255);
  analogWrite(enB, 255);
  digitalWrite(motor1A, HIGH);
  digitalWrite(motor1B, LOW);
  digitalWrite(motor2A, HIGH);
  digitalWrite(motor2B, LOW);
}

void backward() {
  analogWrite(enA, 255);
  analogWrite(enB, 255);
  digitalWrite(motor1A, LOW);
  digitalWrite(motor1B, HIGH);
  digitalWrite(motor2A, LOW);
  digitalWrite(motor2B, HIGH);
}

void left() {
  analogWrite(enA, 255);
  digitalWrite(motor1A, HIGH);
  digitalWrite(motor1B, LOW);
  analogWrite(enB, 255);
  digitalWrite(motor2A, LOW);
  digitalWrite(motor2B, HIGH);
}

void right() {
  analogWrite(enA, 255);
  digitalWrite(motor1A, LOW);
  digitalWrite(motor1B, HIGH);
  analogWrite(enB, 255);
  digitalWrite(motor2A, HIGH);
  digitalWrite(motor2B, LOW);
}

void stopMotorsDrive() {
  analogWrite(enA, 0);
  analogWrite(enB, 0);
  digitalWrite(motor1A, LOW);
  digitalWrite(motor1B, LOW);
  digitalWrite(motor2A, LOW);
  digitalWrite(motor2B, LOW);
}
```