International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Surveillance Car Bot Using AI Thinker-Esp32 Cam

Dr. S. Ananth[1], Soundharya S[2], Tamilarasan R[3]

[1,2,3]Head of the Department, UG Scholars(B-Tech), Department of Artificial Intelligence and Data Science, Mahendra Engineering College Namakkal

Abstract:
The creation of affordable, effective surveillance systems has grown in significance as a means of improving security in a variety of situations, including homes, workplaces, and industrial facilities. The ESP32-CAM module, a low-cost microcontroller with built-in Wi-Fi and Bluetooth capabilities, is used in this project's surveillance bot to construct a small, wireless, and real-time monitoring system. The core of the bot's operation is the ESP32-CAM module, which has a camera and computing power and allows for remote video streaming and monitoring via a web-based or mobile interface. The surveillance bot is made to be extremely adaptable. It can move about the monitoring area thanks to a simple robotic platform made up of motors and a motor driver. A live video feed is captured by the bot's camera and wirelessly sent via a Wi-Fi network to a device that is connected. The system incorporates object detection utilizing the YOLO (You Only Look Once) algorithm to improve surveillance capabilities. This allows for the real-time tracking and identification of objects inside the recorded video feed. An effective method for intelligent monitoring that doesn't require a lot of hardware is to use the ESP32-CAM for image processing and capture in conjunction with YOLO-based object detection. With an easy-to-use mobile or web application that allows for remote control and bot monitoring, the system is made to be user-friendly. This project demonstrates the ESP32-CAM's potential as a flexible and reasonably priced platform for creating intelligent surveillance systems that strikes a balance between cost, performance, and ease of use.

Keywords: battery, SD card, Arduino IDE, Open CV, ESP32-Camera Module, Arduino UNO/FTDI Programmer, DC Motors, Motor Driver (L293D), and ESP IDF.

1. INTRODUCTION

Using the ESP32-CAM module, the Surveillance Bot is an affordable, easy-to-use, and environmentally responsible solution for real-time video recording, storing, and monitoring. With its integrated Wi-Fi and high-resolution OV2640 camera sensor, it facilitates wireless data transfer for smooth streaming to web interfaces or mobile apps. Utilizing open-source software and Slightweight libraries for Internet of Things connectivity, the system integrates local storage on a microSD card with optional cloud upload for video inspection. Its user-friendly interfaces make monitoring and management simple, and its minimal setup and maintenance requirements guarantee scalability for networked operations. It is a cost-effective and useful substitute for conventional surveillance systems because of its small size and robust construction, which enable deployment in a variety of environmental circumstances.

2. SYSTEM STUDY

A conventional surveillance system, usually stationary, that records and captures video. Delays may be introduced during recording and playback for real-time monitoring. Particularly for larger systems with numerous cameras and recording devices, it may be more costly.

2.1 CLASSICAL CBI SYSTEMS

- **Hardware Configuration:** Closed-circuit television (CCTV) cameras are linked to a central network video recorder (NVR) or digital video recorder (DVR) for traditional surveillance.
- **Wired Infrastructure:** The majority of CCTV systems need a lot of cable to transmit data and provide power, which can be expensive and difficult to install.
- **Cost:** It is not affordable for small-scale or individual applications because to the comparatively high initial equipment, installation, and maintenance costs.

2.2 IMPLICATIONS IN THE CURRENT SYSTEM

- **Cost and Accessibility:** These systems are only available to larger businesses or wealthy households due to their high setup and maintenance expenses. User customization and flexibility are restricted by proprietary software and designs.
- **Scalability:** Increasing coverage or adding more cameras frequently requires a large investment of time and money.
- **Dependency on Proprietary Systems:** Users of many current systems are forced to use particular software or hardware ecosystems. Usability may be restricted by incompatibility with other devices.
- **Energy consumption:** Because traditional surveillance systems frequently use a lot of electricity, they are not appropriate for off-grid or distant applications.

3. PROPOSED SYSTEM

The suggested surveillance robot with the ESP32-CAM module provides an affordable and easy-to-use solution for tracking, object detection, and remote monitoring. The robot's sophisticated features and small size allow for real-time video recording and analysis via an easy-to-use interface. Key parts such the ESP32-CAM module, motor drivers, motors, wheels, chassis, and battery are assembled as part of the hardware setup to create a mobile platform that can navigate a variety of settings. Using facial recognition, the system compares a person's face at the entrance with For further security, a list of approved faces is kept on the SD card of the ESP32-CAM.

The Arduino IDE and ESP32-CAM library are used to program the system, which uses OpenCV for image and video processing. This enables real-time video feed capture and analysis while the robot maintains control over its movement. The surveillance robot is a useful and adaptable solution for home, business, and industrial applications because of its low cost, simplicity of use, and sophisticated features like object tracking and facial recognition.

3.1 PROBLEM DEFINITION

The cost, installation, and operation of traditional surveillance systems are all quite difficult. Because they frequently demand for costly hardware, expert setup, and substantial wiring, they are less affordable for individuals, small enterprises, and corporations with tight budgets. Additionally, these technologies are not portable or adaptable, which limits their application in stationary settings and reduces their efficacy in changing circumstances. Many people are left without a practical way to effectively and efficiently monitor their environment due to this high barrier to entry.

Furthermore, sophisticated features like object detection, facial recognition, and mobility—all of which are becoming more and more crucial in contemporary security applications—are usually absent from traditional surveillance systems. By detecting threats, identifying authorized users, and tracking items or people in real time, these features are crucial for improving security. However, people with little technical knowledge may be put off by the complicated configurations, high costs, and dependence on specialized software required
to integrate such capabilities into conventional systems.

A cost-effective, portable, and user-friendly surveillance system that incorporates contemporary features like real-time video streaming, object detection, facial recognition, and mobility is desperately needed to meet these problems. For a system to serve a broad spectrum of users, including people with little technical expertise, it must be simple to set up, implement, and run. The suggested method seeks to bridge the gap between affordability and sophisticated capability by offering a flexible and approachable substitute for conventional surveillance systems by utilizing small hardware, such as the ESP32-CAM, and open-source programming and processing tools.

3.2 OBJECTIVE OF PROPOSED SYSTEM

The main goal of the suggested surveillance robot using the ESP32-CAM module is to create an affordable, portable, and easy-to-use system for security and real-time monitoring. The system's goal is to offer a cost-effective solution that incorporates cutting-edge capabilities like mobility, object detection, and facial recognition, making it appropriate for a variety of uses in residential, business, and industrial settings.

By enabling real-time video streaming, object tracking, and facial recognition for authorized users, the system aims to improve security. The project intends to provide effective and dependable monitoring functionalities while guaranteeing low maintenance and ease of use by utilizing the ESP32-CAM's capabilities and open-source technologies like OpenCV and the Arduino IDE. Additionally, by creating a plug-and-play setup with less configuration, the suggested approach seeks to make deployment and usability simpler. The objective is to make sophisticated surveillance technologies available to non-technical people so they may safely and remotely monitor their environment without needing a lot of technical know-how or resources.

3.3 FEATURES OF PROPOSED SYSTEM

- Real-Time Video Streaming.
- Cost-Effectiveness.
- Compact and Wireless.
- Energy Efficiency.
- Ease of Deployment.

4. Scalable Design.

4. SYSTEM SPECIFICATION

The technical and functional specifications for the Surveillance Bot that uses the ESP32-CAM module are described in the system specification. This document acts as a guide for the system's conception, creation, and deployment, guaranteeing that every part functions as a whole to meet the project's goals.

4.1 ESSENTIAL HARDWARE NEEDS

The essential parts required to properly install and run the surveillance system, with an exclusive emphasis

on real-time video collection, storage, and streaming, are known as minimum hardware requirements. These elements guarantee the system's usability, robustness, and accessibility for applications involving remote monitoring.

ESP32-CAM Module

- FTDI Programmer
- DC Motor
- Motor Driver
- Battery
- SD Card

4.1.1 ESP32-CAM

Based on the ESP32, the ESP32-CAM is a tiny, low-power camera module. Along with an integrated TF card slot, it has an OV2640 camera. Intelligent Internet of Things applications including wireless video monitoring, Wi-Fi picture upload, QR identification, and more can make extensive use of the ESP32-CAM. A small, inexpensive, and multipurpose development board, the ESP32-CAM combines a camera module with the ESP32 microcontroller. It is frequently utilized in projects that call for real-time video streaming, picture capture, or computer vision features. It was created for Internet of Things and embedded systems applications.

4.1.2 PROGRAMMER FOR FTDI

By connecting an FTDI port to a computer's USB port, an FTDI programmer enables users to program or transfer data between the two devices. Among the applications for an FTDI programmer

- Microcontroller programming,
- Arduino Pro Mini programming STC, NXP, Renesas and NEC MCU burning,
- Data exchange between microcontrollers and computers
- GPS device connectivity

4.1.3THE ARDUINO UNO

The Arduino UNO is an open-source, programmable microcontroller board that is inexpensive, versatile, and simple to use. It may be used in a wide range of electronic projects. In addition to controlling relays, LEDs, servos, and motors as an output, this board can communicate with other Arduino boards, Arduino shields, and Raspberry Pi boards. The open-source electronics platform Arduino is built on user-friendly hardware and software. Arduino boards have the ability to take inputs, such as a light on a sensor, a finger on a button, or a tweet, and convert them into outputs, such as turning on an LED, publishing something online, or turning on a motor.

4.1.4 DC MOTOR

Small DC motors are utilized in a variety of home appliances, toys, and gadgets. The applications in retail Conveyors and turntables are examples of DC motors, and huge DC motors are also used for braking and reversing in industrial settings. A DC motor is a device that transforms direct electricity into mechanical work. "Faraday's law of electromagnetic induction" is the foundation upon which the DC motor operates. "Whenever a current-carrying conductor is placed in a magnetic field, it experiences a force," according to Faraday's law of electromagnetic induction.

4.1.5 MOTOR DRIVER

Two DC motors can be controlled concurrently in any direction by the 16-pin L293D motor driver integrated circuit. Up to 600 mA (per channel) of bidirectional drive current may be provided by the L293D at voltages ranging from 4.5 V to 36 V (at pin 8!). To ensure dependable and effective functioning, motor

drivers are employed to control a motor's speed, direction, and occasionally other parameters. It is used to regulate the direction and speed of a small DC motor, or two motors if you want. It is powered by a separate current source that is concealed by a power supply module, but it receives the majority of the signals that operate the motor from the Arduino. Motor drivers employ a little voltage signal from a microcontroller or control system to provide the motor with high power. The motor driver will rotate the motor in a single direction while maintaining one pin as HIGH and one pin as LOW if the CPU sends a HIGH input to the driver.

4.1.6 BATTERY

A battery is a device that uses an electrochemical oxidation-reduction (redox) reaction to directly transform the chemical energy found in its active components into electric energy. Electrons are transferred from one material to another through an electric circuit in this kind of reaction**.**

4.1.7 SD CARD

Videos and pictures can be loaded onto the SD card. Applications can be installed on the SD card. The SD card cannot be moved from one device to another. You can use the SD card in addition to the storage on your device. Secure digital cards, or SD cards for short, are a kind of detachable storage device that is used to store and move digital data. Electronic gadgets like digital cameras, smartphones, tablets, and portable game consoles frequently use it.

4.2 ESSENTIAL SOFTWARE NEEDS

Defining the resources and prerequisites required for an application to run as efficiently as possible on a computer is known as software requirements. Usually not included in the program installation package, these prerequisites need to be installed independently before the product can be installed.

OPERATING SYSTEM-WINDOWS 10 & 11

An operating system (OS) is a program that manages all other application programs on the computer once it has been first loaded by a boot program. Through a designated application program interface (API), application programs communicate with the operating system by requesting services. Software requirements include outlining the prerequisites and software resources that must be installed on a computer in order for an application to operate as best it can. Usually not included in the software installation package, these prerequisites or requirements need to be installed individually before the product can be installed.

- ESP32-CAM Libraries
- ESP32 Board Package
- Arduino IDE
- Web Interface Framework
- WebSocket or HTTP protocol
- ESP32-CAM Drivers
- Code Editor (such as Visual Studio Code) (Optional)

4.2.1 ARDUINO IDE

A software platform called the Arduino Integrated Development Environment (IDE) is used to write, compile, and upload code to Arduino boards, such as the ESP32 microcontroller. It offers a straightforward and easy-to-use interface for hardware project development. You may use the Arduino IDE to create C or C++ scripts that communicate with the hardware, including motors, sensors, and cameras, and then upload

the code to the ESP32 board. This IDE is frequently used in IoT-based applications like surveillance bots and is especially helpful for programming embedded systems.

4.2.2PACKAGE FOR THE ESP32 BOARD

The Arduino IDE can interact with and program ESP32-based boards thanks to a collection of tools and libraries called the ESP32 Board Package. The ESP32 is a potent microcontroller with Bluetooth and Wi-Fi, which makes it appropriate for Internet of Things applications like surveillance. Installing the ESP32 Board Package, which supports the ESP32 architecture, peripheral functions, and other features unique to this board, is required in order to program an ESP32 using the Arduino IDE. This makes it possible for you to create applications for motion detection,
live video streaming, and other surveillance-related duties.

4.2.3 LIBRARIES FOR ESP32 CAMERAS

Specialized libraries known as ESP32-CAM Libraries offer user-friendly features for interacting with the camera module on the ESP32-CAM board. These libraries facilitate the processing and manipulation of picture data in addition to the capture and streaming of photos or movies from the camera. The libraries contain routines for setting up the camera, sending video frames to a server or web interface via a network connection, and initializing the camera. These libraries let you easily create surveillance systems that record in real time and carry out functions like facial recognition and motion detection.

4.2.4 OPENCV

An open-source library called OpenCV (Open Source Computer Vision Library) was created for applications involving real-time image processing and computer vision. It offers a wide range of features and methods for processing video streams, object detection, facial recognition, and other applications. The ESP32-CAM module's video may be analyzed using OpenCV, which then applies machine learning models to identify particular events or objects, such cars or people. OpenCV can greatly improve your bot's capabilities by providing sophisticated image processing functions, while it is not necessary for all surveillance systems**.**

4.2.5 INTERFACE FRAMEWORK FOR WEB

The resources required to develop an intuitive front-end for communicating with the surveillance bot are provided by a web interface framework. A dashboard to examine camera status, show live video feeds, and adjust the bot's settings can be part of this. Flask (for Python), Node.js, or even React for creating the interactive elements are popular web frameworks for these kinds of activities. A web interface framework provides the resources needed to create an easy-to-use front-end for interacting with the surveillance bot. This can include a dashboard to check camera status, display live video feeds, and modify the bot's settings. Popular web frameworks for these kinds of tasks include Flask (for Python), Node.js, or even React for constructing the interactive parts. The framework allows you to create responsive web pages that can communicate with the bot, give commands (such as to begin or stop recording), and deliver real-time information (such as motion alerts or system health).

4.2.6 WEB SOCKET or HTTP PROTOCOL

The surveillance bot and a web interface exchange data using the WebSocket and HTTP communication protocols. Requests between a client (such as a web browser) and a server, including obtaining video streams or status updates from the bot, are frequently made using HTTP. The more effective protocol for real-time communication is WebSocket, which keeps the connection open between the client and server so that data can be sent as soon as it becomes available. While HTTP can be used for user requests and

periodic data updates, WebSocket is particularly helpful for surveillance bots when it comes to streaming live video or generating real-time warnings.

4.2.7 DRIVERS for ESP32 CAM

The software components known as drivers enable communication between the ESP32-CAM board and your computer or development environment. The Arduino IDE can identify the ESP32-CAM board and upload the code to the microcontroller thanks to these drivers. To guarantee correct communication between the computer and the ESP32-CAM, the required drivers must be loaded, depending on your operating system (Windows, macOS, or Linux). You won't be able to upload code or solve problems efficiently without these drivers.

4.3 PACKAGES AND LIBRARIES

A package is made up of several linked modules intended to work together and provide particular features. Like independent modules, these modules can be imported and are organized within a directory. There might be sub-packages in this directory, each of which might include other modules buried inside still more subdirectories. On the other hand, the term "library" is more wide and typically refers to "a collection of code." Libraries can contain many separate modules, frequently offering a wide variety of features. The Standard Library, for example, has hundreds of modules designed for everyday activities like emailing or parsing JSON data. The Standard Library stands out since it comes with your programming language installation, allowing you to use its modules without having to obtain them individually from outside sources. This integration improves accessibility to key features for developers and expedites the development process.

5.SYSTEM DESIGN

5.1 INTRODUCTION

A data flow diagram (DFD) is a graphic depiction that shows how data flows through a system using standardized symbols and notations to indicate how a business runs. Formal approaches such as the Structured Systems Analysis and Design Method (SSADM) frequently include these diagrams. A DFD shows how data moves through a system or process, usually an information system. It also describes the process itself, as well as the inputs and outputs of each entity. DFDs don't show loops, decision rules, or control flow like flowcharts do. Rather, they concentrate on the flow of data. Using a flowchart, specific operations based on the data can be shown. DFDs fall under the category of structured analytical modeling tools and can be displayed in a variety of ways. They are well-liked because they make the important facts and actions in software-system processes easier to see.

CONTEXT DIAGRAM

One special kind of data flow diagram (DFD) that represents a whole system as a single process is called a context diagram. The interactions between the system and outside entities are highlighted. A high-level visual depiction of a system and its relationships with other things is called a context diagram. It displays the system as a single process and emphasizes the information flow between the system and its external components, including databases, users, and other systems. A context diagram's objective is to offer a straightforward, uncomplicated illustration of the system's limits and how it interacts with the environment.

5.2 PRIMITIVE SYMBOLS

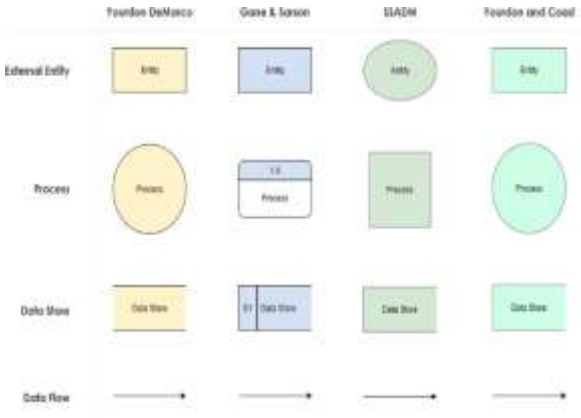


Fig1:Primitive Symbols

Symbols of DFD are:

- External Entity
- Process
- Data Store
- Data flow

5.2.1 PROCESS

Processes are essential operations that use information and are carried out inside the system boundary. Only when the data used as input for the activity is changed or transformed, sending different data out of the process than what entered it, is a process represented in the model.

5.2.2 DATAFLOW:

A DFD's data flows show how information moves between various elements, including processes, data stores, and external entities. Arrows linking these elements serve as a visual representation of these flows, and labels identifying the kind of data being transferred are included. They show the routes that data takes when it enters, exits, or passes via the system. Data flows are used to illustrate how information moves through a system and show how it is used and shared. DFDs help stakeholders comprehend data communication and movement throughout the system by providing an illustration of data flows, which helps to clarify the links between components and the data they interact with.

5.2.3 DATA STORE:

A data store, represented by a rectangle with a label in a DFD, is a repository or storage location where data is kept in the system permanently. These stores stand in for databases, file systems, or any other system that maintains information for later access or use. They are essential for showing the locations of data storage over time periods or between activities. For example, a data store in an inventory management system could refer to a database that contains product information. Data stores highlight the permanence and durability of data by helping to model its accessibility and maintenance inside the system.

5.2.4 OUTSIDE ENTITY:

In DFD notation, an external entity—also called a "terminator"—designates external data sources or destinations inside a system. These entities typically interact with the system being studied despite existing outside of it. External entities can be physical objects, people, organizations, or other systems. They are shown as rectangles with appropriate labels in DFDs. Customers, suppliers, and regulatory agencies, for example, could all be considered external entities in a retail system. They are shown as rectangles with appropriate labels in DFDs. Customers, suppliers, and regulatory agencies, for example, could all be

considered external entities in a retail system. By sending or receiving inputs or outputs, these entities communicate with the system.

DFDs use outside parties to show how data enters and exits the system, which helps define its boundaries and interfaces with outside parties.

5.3 CIRCUIT DIAGRAM

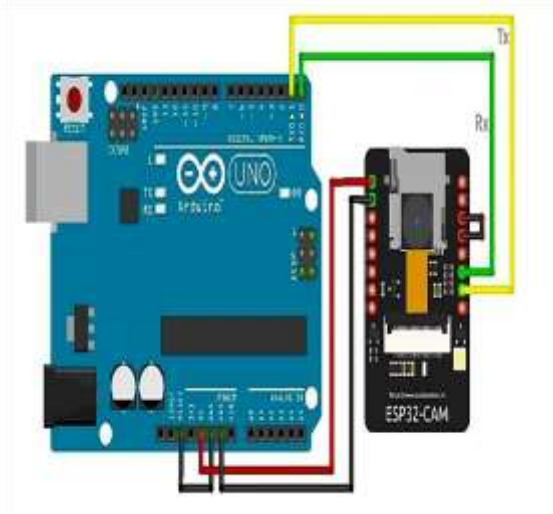


Fig2: Circuit Diagram

6. SYSTEM DEVELOPMENT

6.1 THE IMPLEMENTATION OF HARDWARE

The surveillance bot project's hardware implementation is essential to guaranteeing robustness and dependable operation. To create a coherent system, it entails integrating a number of parts, including the chassis, power supply, sensors, and ESP32-CAM module. A more thorough description of each part and the procedures for constructing it may be found below.

6.1.1 ESP32 CAM-MODULE

Power Supply: For optimal performance, the ESP32-CAM needs a steady 5V power source. A USB adapter or a specialized power source that provides the necessary voltage can be used to supply this. Consistent voltage and current from the power supply are necessary to guarantee uninterrupted ESP32-CAM operation.

Programming and Configuration: A USB-to-TTL converter is required in order to program the ESP32-CAM. With this adapter, you can use a serial interface to upload code to the ESP32-CAM. During programming, connect the USB-to-TTL adapter to the ESP32-CAM's GPIO pins to enable communication. Make that the TX, RX, and other required pins are wired correctly between the adapter and the ESP32-CAM.

Camera Sensor: The OV2640 camera sensor is included with the ESP32-CAM. For clear and precise video to be captured, the camera sensor and module must be properly aligned. After connecting, the camera records video frames, which are then processed and sent to the web interface over Wi-Fi for in-the-moment monitoring.

6.1.2 MICRO SD CARD

Putting the MicroSD Card in: Place a microSD card into the ESP32-CAM module's specific microSD card slot. To prevent connectivity problems, make sure the card is positioned correctly. Use a file system, such

FAT32, that the ESP32-CAM supports to format the card. To make sure there is adequate room for recorded video, it is also advised to verify the storage capacity, which is typically up to 32GB. Storage Management: When motion is detected or the bot is actively streaming video, the video can be stored on the microSD card. Depending on the application, you can set up features that allow you to keep video for a predetermined amount of time or handle files by looping or automatically deleting them when the card is full.

6.1.3 SENSORS

Including a PIR Motion Sensor: The PIR (Passive Infrared) sensor picks up infrared rays from moving objects in its area of vision, including people or animals. Connecting the PIR sensor's output to one of the ESP32-CAM's GPIO pins is necessary for integration. The PIR sensor delivers a signal when it detects motion, which might cause things like warnings or video recording.

Installing the Sensor: The sensor should be mounted so that it can adequately cover the region of interest. If the bot is going to be outside or in a hallway, make sure the sensor is unhindered to precisely detect motion. If necessary, adjust the sensitivity of the sensor to avoid false triggers (e.g., sensing pets or minor movements).

6.1.4 POWER SUPPLY

Power Options: For interior installations, a USB converter is a practical way to supply power to the ESP32-CAM and additional parts. Rechargeable battery packs, such as Li-ion or Li-Po, can supply power for remote or mobile applications. Make sure the battery has adequate capacity to last for a long time between charges. An environmentally responsible way to fuel the bot for outside deployments is with a solar panel. To ensure continuous functioning day and night, a solar panel with an integrated charge controller can assist in controlling the battery's charging.

Power Management: To guard against short circuits, undervoltage, and overvoltage, use a power management module. This guarantees that the parts are not harmed and receive steady electricity**.**

6.1.5 ENCLOSURE

Weatherproofing for Outdoor Use: It is essential to put the components of the surveillance bot in a weatherproof housing if it is to be used outdoors. By doing this, damage from rain, snow, or extremely high or low temperatures is avoided. Additionally, the case needs to be strong enough to resist damage or physical knocks.

Material Selection:

- **Plastic/ABS Enclosure:** Perfect for safeguarding electronics in mild outside environments, plastic enclosures are lightweight and simple to produce.
- **Metal Enclosure**: Metal enclosures offer superior protection against electromagnetic interference (EMI) and physical damage, particularly in hostile situations.

Mounting Options: To firmly fasten the enclosure to a wall, ceiling, or pole, it needs have mounting points or brackets. Make sure the camera lens can capture the necessary area of interest and has an unhindered view.

6.1.6 CHASSIS CONSTRUCTION

Assembling the Chassis: The chassis can be built using 3D-printed components, acrylic, or plastic. These materials are sufficiently robust and lightweight to hold the components in place. All hardware parts, including the ESP32-CAM, sensors, battery, and motor drivers, should be able to be mounted or slotted into the chassis.

Customization: You can customize the chassis with characteristics like wheels or legs for mobility or fixed installation for a stationary arrangement, depending on the use case. For ease of placing in a variety of settings, such as on a shelf, wall, or ceiling, the chassis should be small.

6.1.7 COMPONENT MOUNTING

When mounting the ESP32-CAM, make sure the camera has an unhindered view of the area that needs to be watched. To firmly attach the module to the chassis, use mounts, screws, or zip ties**.**

Mounting the Sensors and Other Components: Make sure the PIR motion sensor is positioned at the proper height and angle to detect motion. Install any additional sensors (such as humidity or temperature sensors) in accordance with the region you wish to keep an eye on.

Mounting of the Battery and Motor: Place the battery pack in a convenient location for replacement or charging. Make sure the motors or motor drivers are placed on the chassis and linked to the ESP32-CAM for movement control if they are utilized for mobility.

6.1.8 POWERING AND LINKING

Connecting Components: To connect components, use jumper wires, male-to-female connectors, or specially made PCBs. To prevent miswiring, make sure the pins are aligned correctly. Through its GPIO ports, the ESP32-CAM can be connected to external sensors, such as the PIR sensor. If necessary, use the proper voltage regulators or resistors**.**

Maintaining Correct Polarity and Connections: To prevent component damage, make sure that the positive and negative connections are appropriately matched. This covers the ESP32-CAM, sensors, motors, and battery power connections. To prevent sporadic power problems, ensure that the power lines are firmly attached for stability.

Cable management: To keep wires neat and avoid tangling, use cable ties, clips, or sleeves. This lowers the possibility of short circuits and keeps the system tidy.

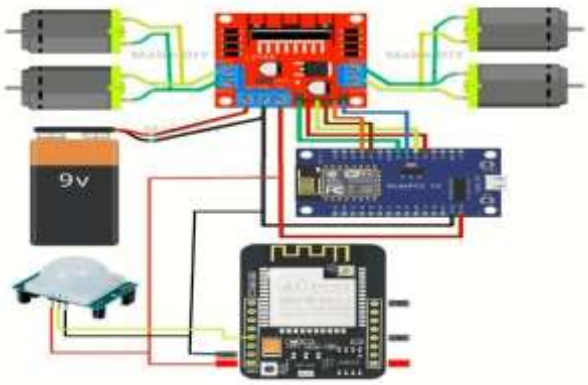


Fig3: Advanced Circuit Diagram

6.2 SETTING UP SOFTWARE

Programming Environment: To write firmware, use the ESP-IDF or Arduino IDE. Install the necessary libraries, such as the HTTP server, Wi-Fi, and ESP32 camera libraries.

Essential Features: Set up the camera and adjust parameters like frame rate and resolution. Configure the web server to handle system configurations and broadcast video. Incorporate techniques for facial recognition or motion detection.

Firmware Testing: Use the USB-to-TTL converter to upload the firmware to the ESP32-CAM. Verify that the camera is operational by making sure it records footage and reacts to commands.

Firmware Development: Use the Arduino IDE or a comparable development environment to write the ESP32-CAM's firmware code. To communicate with the different hardware components, implement the hardware abstraction layer (HAL). Create device drivers for every part (camera, sensors, and motors). Put sensor processing algorithms (such filtering and calibration) into practice. Create control and navigation algorithms (e.g., obstacle avoidance, line following). Put communication protocols into practice to send data wirelessly. Construct the user interface for data visualization and remote control.

Testing and Debugging: To find and address any problems, thoroughly test each software module separately. To fix any mistakes or strange behavior, debug the system.

6.3 DEVELOPMENT OF WEB APPS

Design and Develop the User Interface: Provide an intuitive user interface for data visualization and remote control.

Put User Interactions into Practice: Give consumers the ability to change camera settings, move the robot, and watch live video streams. Create a web interface that is lightweight by utilizing HTML, CSS, and JavaScript.

Design: The procedure for producing the app's interaction and visual components, including color schemes, icons, and images Use the internal server of the ESP32 to host the web interface in order to offer: real-time video stream. Configurations of the system, such as Wi-Fi settings and motion sensitivity. Options to start and stop recording

MOBILE APPLICATION(OPTIONAL)

Create a mobile application with a user-friendly interface for managing the bot and reading its feed by utilizing frameworks such as Flutter or React Native.

Cloud Integration: To upload video, set up a cloud storage platform such as Firebase or AWS. Use protocols such as REST APIs or MQTT to manage IoT features and deliver notifications**.**

6.4 WORKFLOW IMPLEMENTATION

6.4.1. INITIALIZATION

Turn on the ESP32-CAM: The system begins by turning on the ESP32-CAM module, which can be done via a USB connection to a power source or by utilizing a battery that has been properly voltage-regulated. The camera module, Wi-Fi interface, and microSD card (if utilized for video storage) are among the onboard systems that the ESP32-CAM initializes.

Establish Wi-Fi Connectivity: The ESP32-CAM uses its integrated Wi-Fi module to connect to a pre-configured Wi-Fi network. In order to ensure that the device has internet connection (if necessary) and is able to communicate with other devices via the network, this step entails connecting the bot to the router.

Verify System Initialization: The system performs a number of initialization tests to make that the camera module is operating properly and that it is capable of recording video after turning on and connecting. Every sensor is operational and prepared to initiate events, such as a PIR for motion detection. Video streaming and web interface interaction are made possible by the setup and operation of all communication protocols, such as HTTP and WebSocket.

6.4.2 STREAMING VIDEOS

Access the Web Interface: By using a browser and inputting the IP address that the router has assigned to the ESP32-CAM, the user can view the live video stream of the surveillance bot. If this IP address is manually assigned, it is static; if it is assigned by the router's DHCP service, it is dynamic. The live video feed appears on a webpage that loads after the IP address is browsed.

The ESP32-CAM records video frames from the camera module, compresses them, and then transmits them over the network via HTTP or WebSocket to confirm real-time video streaming. Users can watch the live footage with little delay because the web interface continuously gets the video stream. To guarantee real-time monitoring, performance testing should be carried out to make sure the video stream functions without noticeable lag or disruption.

6.4.3 ALERTS AND MOTION DETECTION

Introduce Motion into the Camera's Field of View: A PIR sensor or computer vision algorithms (like OpenCV) are commonly used to detect motion in a video feed. In order to identify movement, the PIR sensor looks for variations in the infrared light inside its range of vision. As an alternative, motion can be identified by employing OpenCV-based algorithms to examine the video stream for notable variations between successive frames**.**

Check Motion Detection: The system ought to notify the user or initiate an action (like beginning video recording) when motion is detected. Depending on the configuration selected, the notifications may be sent via email, SMS, or push notification via an app.

Test Notification Functionality: Making sure the notification system is dependable and responsive is a crucial step in this process. To confirm that the system detects motion and provides real-time notifications, test several motion scenarios (such as a person walking in front of the camera or a pet moving nearby). To guarantee prompt replies, make sure the user gets alerts as soon as motion is detected.

6.4.5 OBJECT DETECTION

YOLO Algorithm: Real-time object identification and tracking inside a video feed is made possible by object detection, a crucial component of contemporary surveillance systems. To improve its monitoring capabilities, the Surveillance Bot in this project incorporates the YOLO (You Only Look Once) algorithm. With its reputation for speed and accuracy, YOLO is a deep learning-based object recognition framework that is perfect for real-time applications on devices with limited resources, such as the ESP32-CAM.

ESP32-CAM: This solution uses a lightweight YOLO model that is tailored for embedded systems to handle live video frames that are captured by the ESP32-CAM. Bounding boxes and labels that emphasize the detected items are sent to a web or mobile interface for remote monitoring along with the video feed. For added protection and automation, spotted items may also set off alarms or logging systems.

6.4.6 DATA STORAGE

Record Video on the MicroSD Card: To store video footage, the ESP32-CAM can be connected to a microSD card. locally. The microSD card records the video when motion is detected or when streaming starts. To make it simple to locate particular video, file structure and naming conventions (such as date and time) should be set.

Test Cloud Synchronization for Backup: The system can be set up to synchronize recorded video with a cloud storage service (like Google Drive or AWS S3) in order to guard against data loss. The technology saves files to the cloud for remote access and backup when video is captured. Verify cloud synchronization to make sure the video is consistently uploaded and accessible from a distance if necessary. To prevent going over storage restrictions, make sure that data is synchronized on a regular basis and that cloud storage is maintained.

6.4.7 OBSERVATION AND MANAGEMENT

Remotely Access Live Feed and System Settings: Users can view real-time footage and access the live video stream from any location with internet access by using the web interface or mobile app. Users can

change system parameters like video resolution, camera tilt/pan, and motion detection sensitivity in addition to streaming videos.

Test User Commands: Using the online interface or app, users can control the surveillance system by turning motion detection on or off, adjusting camera resolution, and changing other parameters. User actions should be instantly processed by the system, and changes should be reflected in real time on the interface (for example, altering the video resolution should instantly impact the live stream).

6.4.8 INDOOR DEPLOYMENT

Set up the Bot in Indoor Environment: The surveillance bot can be deployed indoors to monitor areas like rooms, hallways, offices, or entrances. Ensure that the camera's field of view is clear of obstructions and covers the intended monitoring area.

Test Performance in Indoor Environment: Test the system's ability to capture and stream video indoors, ensuring proper lighting conditions and visibility. Adjust settings (e.g., exposure, resolution) based on the indoor environment to get the best video quality.

6.4.9 OUTDOOR DEPLOYMENT

Install the Bot in a Weatherproof Enclosure: To protect the ESP32-CAM from dust, moisture, and temperature changes, it must be placed in a weatherproof enclosure for outdoor deployment. The enclosure should include features that allow the camera to be mounted at the proper angle and provide good visibility.

Test Performance in Outdoor Environment: Ensure that the system works well in outdoor lighting conditions, such as direct sunlight or low-light environments. The camera should be configured for optimal video quality (e.g., adjusting brightness, contrast, or IR settings for night vision). The bot should also be resistant to outdoor elements like rain, wind, and temperature changes.

6.4.10 SCALING

Deploy Multiple ESP32-CAM Units: For larger areas (e.g., buildings, outdoor parks), multiple ESP32-CAM units can be deployed, each covering a different section of the area. These units can be interconnected using a local network, allowing a single web interface to monitor multiple feeds simultaneously.

Test System Scaling: Verify that the system can handle multiple video streams and motion detection events without performance degradation. Ensure that the web interface can display feeds from all deployed units and allow users to switch between different camera views seamlessly.

7.MODULE DESCRIPTION

The Surveillance Bot using ESP32-CAM is divided into several essential modules, each focusing on specific functionalities to ensure a comprehensive and efficient surveillance system. These modules collaborate to offer seamless monitoring, storage, and remote access for enhanced security purposes. The development of the Surveillance Bot using ESP32-CAM involves a structured, multi-stage approach to ensure a reliable, scalable, and effective surveillance system. Each phase of the development process has been carefully planned and executed to ensure maximum efficiency and security.

7.1 REQUIREMENT ANALYSIS

This phase comprises determining the basic aims and functional needs of the surveillance bot. Key tasks include:

- Defining system goals such as real-time video capture, data storage, and remote monitoring.
- Identifying hardware components: ESP32-CAM module, microSD card, power supply, motor drivers, and optional sensors.

- Outlining software tools required, including the Arduino IDE, ESP32 board libraries, and optional OpenCV for face recognition.
- Planning for power management, wireless communication, and user interface needs.

7.2 HARDWARE PROCUREMENT and ASSEMBLY

This stage focuses on sourcing the necessary components and assembling the hardware. Steps include:

- Procuring ESP32-CAM modules, power supply units, microSD cards, and additional components.
- Assembling the ESP32-CAM module on a stable chassis.
- Integrating microSD storage and ensuring proper connectivity with the camera module.
- If mobility is required, adding motor drivers, wheels, and proximity sensors.
- Ensuring proper wiring and circuit protection for safe operation.

7.2.1 CAMERA MODULE

The camera module is the core component responsible for video capture and transmission using the OV2640 camera sensor integrated with the ESP32-CAM Key functionalities include:

- High-resolution video capture with adjustable frame rates for balancing quality and bandwidth.
- Real-time streaming over Wi-Fi, making the feed accessible remotely via a web interface.
- Automatic adjustments for exposure and white balance, ensuring optimal video quality under varying lighting conditions.
- Image compression for efficient storage and transmission, reducing data usage.
- Customizable settings for different surveillance needs such as indoor and outdoor monitoring.

7.2.2 STORAGE MODULE

In order to ensure appropriate data management, the storage module manages the local saving of recorded video and pictures on a microSD card. Important characteristics include:

- Automatic file management with timestamping for simple organization and retrieval
- Data optimization techniques to prevent storage overflow and automatic deletion of older files
- Real-time storage of image snapshots and video feeds
- Compatibility with different microSD card capacities for flexible storage options.

7.2.3 COMMUNICATION MODULE

The communication module enables remote access and data transfer between the surveillance bot and user devices. Core functionalities include:

- Built-in Wi-Fi support for streaming video feeds directly to a web-based interface.
- Hosting a web server for secure remote access using IP-based connections.
- Password protection and encryption for secure data transmission.
- Ability to stream live footage to multiple devices simultaneously without lag.

7.2.4 MODULE FOR POWER MANAGEMENT

Rechargeable Li-ion batteries and 5V adapters are integrated for dependable power supply.

- Voltage regulation to protect sensitive components from power fluctuations.
- Low-power modes to conserve battery life when the system is idle.
- Battery monitoring and alerts to prevent unexpected shutdowns.

7.2.5 USER INTERFACE MODULE

A web-based control panel for viewing and managing surveillance footage is provided by the user interface module. Features include

- Standard web browsers can view real-time live streaming.
- Zoom, focus, and brightness adjustments are available for the camera.
- Password protection and user authentication ensure secure access.
- Mobile-friendly design allows monitoring from smartphones and tablets.

7.2.6 MOTION CONTROL MODULE

The motion control module gives the surveillance bot mobility so it can move around on its own. Important characteristics include:

- Integration of motor drivers for directed motion.
- Detecting obstacles with proximity sensors to prevent collisions.
- Control movement in real time through the web interface.
- Automated area coverage with programmable patrol routes.

7.3 SOFTWARE DEVELOPMENT

Writing and uploading the required code to the ESP32-CAM module is the main goal of the software development phase. Important jobs consist of:
Setting up the ESP32-CAM board via the Arduino IDE.
Writing code for Wi-Fi connectivity, real-time streaming, and video capturing.
Implementing the microSD card's data storing capabilities.

- Creating a simple web-based user interface for remote access.
- Using OpenCV and image processing tools for optional facial recognition.

7.4 FUNCTIONAL TESTING AND INTEGRATION

Integration and testing are carried out to confirm system performance after the hardware and software components are ready. The tasks consist of:

- Connecting the ESP32-CAM to the communication modules, power supply, and microSD card.nTesting frame rates, resolution, and video capture quality.
- Assessing the dependability of distant streaming and Wi-Fi connectivity. Testing data storage capacity and power management under duress. Finding and fixing problems with hardware-software compatibility.

7.5 IMPLEMENTATION AND EFFICIENCY

In order to verify practical usefulness, this phase entails deploying the surveillance bot in a real-world testing setting. Important actions consist of:

- Testing the video quality by deploying the bot in various indoor and outdoor settings.
- Adjusting frame rates and camera resolution according to performance.
- Optimizing data compression and Wi-Fi signal strength.
- Dealing with problems like heat management and battery life optimization.

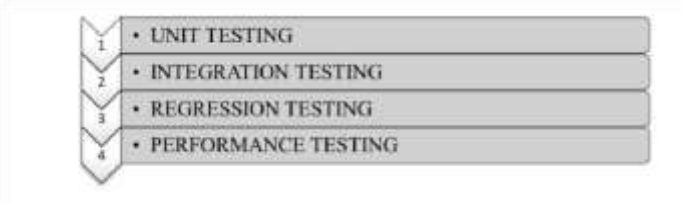
7.6 UPKEEP AND RECORD KEEPING

Following deployment, the emphasis switches to making sure that maintenance is simple and reliable over the long run. The tasks consist of:

- Creating thorough instructions for software installation, hardware configuration, and troubleshooting.
- Offering instructions to non-technical people so they can utilize the system.
- Preparing recurring firmware upgrades for new features and security patches.

8.SYSTEM TESTING

8.1 TYPES OF TESTING:



8.1.1 UNIT TESTING

A key component of the Smart Guided Glass testing procedure is unit testing, which concentrates on the system's smallest, independently testable parts. These elements may consist of separate modules or functions that are in charge of particular architectural duties. Verifying that each of these parts operates as intended when used alone is the main goal in order to guarantee the system's dependability.

8.1.2 TESTING FOR INTEGRATION

Because of the system's complexity, integration testing is essential to assess how various parts—from text-to-speech conversion to obstacle detection—interact and work together when combined to form a coherent whole. It guarantees that every essential component functions in unison, enhancing the system's efficacy. Integration testing comprises assessing the channels of communication between various modules.

For instance, several obstacle recognition algorithms efficiently send obstacle information to the text-to-speech feature, guaranteeing the visually impaired user receives notifications on time. Additionally, it guarantees that text-to- speech features smoothly convey the information to the user.

For the Smart Guided Glass system to operate accurately and dependably, integration testing is necessary.

8.1.3 TESTING FOR REGRESSION

Regression testing is essential to preserving the Smart Guided Glass system's stability and dependability, especially as it develops over time. This testing procedure makes sure that any upgrades, modifications, or additions to the system don't cause new problems or jeopardize already-existing features. When vital components like the text-to-speech conversion module or object detection algorithms are modified, regression testing becomes even more crucial. Regression testing ensures that the system's essential features are maintained and that modifications don't cause unforeseen problems. Instilling confidence in the system's effectiveness and dependability, this method reassures visually impaired users that the technology has undergone extensive testing and validation**.**

8.1.4 PERFORMANCE TESTING:

A crucial step in making sure the Smart Guided Glass technology not only correctly detects impediments but also does it quickly and consistently is performance testing. It evaluates how well the system can manage a range of user demands and workloads, ensuring a smooth and effective user experience. Performance testing evaluates how responsive the system is to requests and inputs from users. It guarantees prompt and accurate feedback to visually challenged users. Testing for scalability determines how well the system can adjust to rising user demands, guaranteeing that it continues to be responsive and effective in meeting the needs of an expanding user base. Through thorough performance testing, the Smart Guided Glass system is optimized to provide a dependable and effective user experience even as system demands increase.

CONCLUSION

The ESP32-CAM Surveillance Bot combines cutting-edge technology with ease of use to provide a workable and affordable answer to contemporary security issues. This project makes use of the ESP32-CAM's capabilities to provide features like motion detection, remote monitoring, real-time video streaming, and optional facial recognition, all in a small, expandable package. The idea uses wireless connectivity and modular functionality to overcome the drawbacks of typical surveillance systems, including their high cost, extensive wiring, and lack of customization. Its adaptability is further increased by its integration with IoT principles, which makes it appropriate for a variety of uses, such as office monitoring, home security, and educational initiatives.

The Surveillance Bot's emphasis on usability and deployment simplicity allows customers to establish and run a reliable surveillance system without the need for significant technical know-how. Because of its small size and low power consumption, it is perfect for do-it-yourselfers and developers who are interested in embedded systems and the Internet of Things. To sum up, this project not only demonstrates the ESP32-CAM's potential for developing creative surveillance solutions, but it also acts as a springboard for next developments in Internet of Things-based security systems. The Surveillance Bot, which has room for more improvements like AI-powered analytics and increased sensor integration, is proof that intelligent technology can make daily living safer and more secure.

REFERENCE

1. F. Tang, Y. Ying, J. Wang and Q. Peng, "A novel texture synthesis based algorithm for object removal in photographs", LNCS, vol. 3321, ASIAN2004 ASIAN2004 pp. 248-258.
2. Haritaoglu, D. Harwood and L.S. Davis, "W4: “Real-Time Surveillance of People and their Activities", IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020.vol. 22, pp. 809-830.
3. M.Md Athiq UR Raza Ahamed and Wajid Ahamed, "A Domestic Robot for Security Systems by Video Surveillance using Zigbee Technology", international Journal of Scientific Engineering and Technology, May 2019, vol. 2, no. 5, pp. 448- 453.
4. P Kumar, A. Singhal, S. Mehta and A. Mittal, "Real-time moving object detection algorithm on high resolution using GPUs", Journal of Real-Time Image Processing, International journal of computer application, Mar. 2019, vol. 11, no. 1, pp. 93-109, 2020. vol. 117.
5. P. Vamsi krishna, S.R. Hussain, N. Ramu, P.M. Rao, G. Rohan and B.D.S Teja, "Advanced Raspberry Pi Surveillance (ARS) system", Proceedings of the 2015 Global Conference on Communication Technologies (GCCT), 21–23 April 20, pp. 860-862.
6. Tasleem Mandrupkar and Manisha KumariRupali Mane, "Smart Video Security Surveillance with Mobile Remote Control", International journal of Advanced Research in Computer Science and Software Engineering, 2022, vol. 3, no. 3.
7. Md. Mohsinur Rahman Adnan “Design and Implementation of Smart Navigation System for Visually Impaired”(International Journal of Engineering Trends and Technology (IJETT) – Volume 58 Issue 2 - April 2018)
8. Jinqiang bai-“Smart guiding glasses for visually impaired People in indoor environment” (IEEE journal paper, Vol. 63, No. 3, August 2017).
9. WHO international statistical classification of diseases and related health problems, 10th Revision ICD-10: tabular list World.

10. Paik, H.Y., Lemos,A.L.: Web Service Implementation and Composition Techniques, vol. 256. Springer International Publishing (2017)
11. Technology, S.F.: About the UltraCane. https://www.ultracane.com/about_the_ultracane. Accessed 4 February 2020
12. Degaonkar, S., et al.: A smart walking stick powered by artificial intelligence for the visually impaired. Int. J. Comput. Appl. 178(32), 7–10 (2019)