VOICE-ENABLED REMOTE SURVEILLANCE SYSTEM

Jitendra Bhandarkar
Dept. of Electronics Engineering (Communication),
Priyadarshini College of Engineering, Nagpur, India
mbhandarkar.jitendra@gmail.com

Gouri Halde
Dept. of Electronics Engineering (Communication),
Priyadarshini College of Engineering, Nagpur
gourihalde@gmail.com

Sunita Parihar
Dept. of Electronics Engineering (Communication),
Priyadarshini College of Engineering, Nagpur
sunitaparihar1824@gmail.com

Abstract

In this particular undertaking, the objective is to create a live video streaming system intended for remote surveillance purposes. With the increasing popularity of video surveillance systems, the ability to monitor an entire system from a distant location has become both intriguing and convenient. This project focuses on the development of an embedded, real-time surveillance system utilizing the ESP32 Cam module. The system encompasses various features including intrusion detection through face detection and recognition, fire detection, voice notifications, live video streaming, and portability. The proposed solution presents an economical, voice-guided, and universally accessible surveillance approach that is both effective and straightforward to implement.

Keywords: Remote, Surveillance, ESP32 Cam Module, Camera, PIR Sensor, Voice Alert, Python, Executable File.

Introduction

Digitalization refers to the transformative influence of digital technologies. In the contemporary era, there is an increasing interconnectedness between individuals and objects to fulfil diverse requirements. Today, homeowners seek comprehensive security measures to protect their homes from threats such as theft, fire, and intrusion. One crucial aspect of surveillance systems is the ability to establish remote connections through the cloud for off-site monitoring. In the past, security systems relied on human guards who continuously monitored surveillance screens. However, the remarkable capabilities of video surveillance systems have contributed to their growing popularity.

We are thrilled to introduce a distinctive product with fascinating features, particularly its voice assistance functionality. Our project incorporates image processing techniques to enhance its performance. Image processing entails analysing input images or video frames and extracting relevant image parameters. Through image processing, we can determine factors like person movement, identification of known or unknown individuals, object recognition, and more. Once the analysis is complete, our system provides voice alerts to notify the user about the identified person, object, activity, or movement. Notably, our system is designed to optimize storage space by activating only when motion is detected in front of the camera.

For image processing and software development, we employed Python as a high-level general-purpose programming language. To set up the ESP32 Cam module, we utilized the Arduino IDE and the embedded C language. This cost-effective development board is equipped with a compact camera module, offering versatility and affordability for creating IoT applications based on image processing.

Shifting our focus to the previously proposed system, while researching security surveillance systems, we encountered several research papers that shared common ideas and concepts:

Ruiguan Ge, Zhenfang Shan and Hao Kou,proposed a system An Intelligent Surveillance System, that the system has the characteristics of a timely alarm with high compression and good usability, motion detection algorithm and quantitative media compression technology. The outcome of the simulation experiment confirms that the improved motion detection algorithm is reliable. Personal device can effectively decode and play surveillance video [1].

Saroja Kanta Panda and Sushanta Kumar Sahu, proposed a systemwhere, they create a fully functional, real-time monitoring system. In the future, plan to use OpenCV as a better way to improve the detection algorithm, because it mainly depends on the threshold value. Its mean algorithm was developed to perform absolute conditions and to determine whether a person is authenticated or not [2].

Qinsheng Du, and Jiling Tang, “Design of the ARM Based Remote Surveillance System,”, They created a system where video has been compressed by the MJPEG algorithm and then directly sent to owner, It’s about Image Compression [3].

Rohan Namdeo, Sahil Sharma, Varun Anand and Chanchal Lohi in July 2020 proposed a system “Smart Automated Surveillance System using Raspberry Pi”, where the system will send an alert notification immediately to the owner whenever there is a movement detected, the system will take images at desired time and start recording video [5].

Sruthy. S, S. Yamuna, and Sudhish N. George, in 2017 proposed a system where person detection with face recognition feature and fire detection are the prime features of this system in “An IoT based Active Building Surveillance System using Raspberry Pi”. And the source is IEEE [6].

Singoee Sylvestre Sheshaiin May 2016 proposed a system as “Raspberry pi Based Security System”, In the system, using a PIR sensor motion will detect, using a Pi Camera video capturing will initiate and sending out an alert via e-mail [8].

After conducting a thorough review of the available literature, we observed several commonalities among the developed systems. Many researchers opted for the Raspberry Pi module, despite its higher cost and limited availability in the market. However, a subset of researchers focused on creating cost-effective solutions by utilizing affordable controllers and cameras. These systems often included features such as face recognition and identification, as well as the generation of alerts through email and text messages.

During our investigation, we identified an opportunity to enhance these existing systems by incorporating voice assistance functionality. Voice assistance offers numerous advantages, especially in terms of providing instant access to information. By integrating this feature into our surveillance system, we can promptly issue alerts in the event of intrusions or suspicious activities. Furthermore, voice assistance can be implemented in multiple local languages to ensure ease of understanding for individuals from diverse backgrounds. This portable feature can be conveniently accessed through a smart mobile handset, providing accessibility at any given time.

Proposed Methodology

The essential elements of such systems comprise the ESP32 Camera module and an executable file (.exe) specifically designed for video processing and monitoring. The ESP32 Cam module is a development board that is Arduino-compatible and was originally developed at the Interaction Design Institute Ivrea in Italy. These systems function as intelligent Internet of Things (IoT) solutions. Whenever the camera detects and captures a person or any other object, the system triggers a voice command stored in its database and promptly sends an email alert notification to the owner. Furthermore, the system captures images at predefined time intervals and initiates video recording.

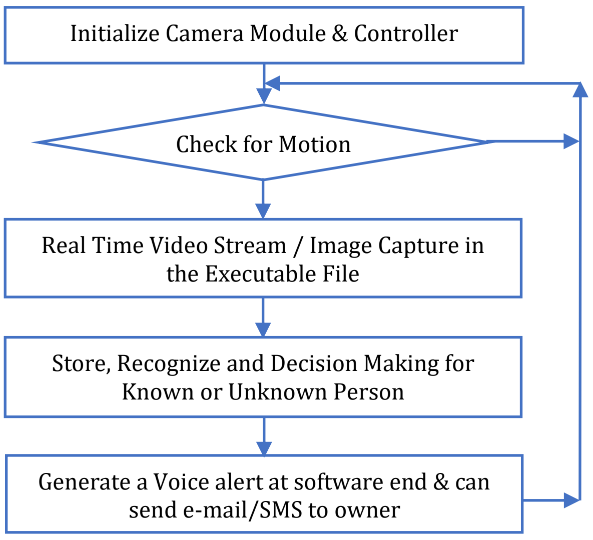

Initialize Camera Module & Controller
Check for Motion
Real Time Video Stream / Image Capture in the Executable File
Store, Recognize and Decision Making for Known or Unknown Person
Generate a Voice alert at software end & can send e-mail/SMS to owner

Figure 1: Flow Chart

The flowchart presented above offers a comprehensive outline of the system's functioning, depicting the various stages involved in motion sensing, image acquisition, and subsequent processing. We have thoroughly examined the implementation process, outlining the essential hardware components and software resources needed for the project's physical realization. During the development of our executable file featuring a Graphical User Interface (GUI), we made effective use of multiple Python software libraries available to us.

Hardware Implementation

The ESP32 CAM Wi-Fi Module Bluetooth incorporates the OV2640 Camera Module, a compact camera module measuring 40 x 27 mm. This versatile module is extensively utilized in IoT applications, thanks to its independent operation capabilities and competitive features. Its wide range of applications includes wireless monitoring, smart home devices, industrial wireless control, and various other IoT applications. With its reliable connection mode, it seamlessly integrates with different IoT hardware terminals.

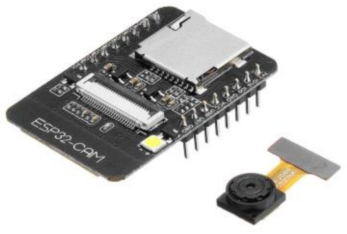


Figure 2: ESP32 Cam Module

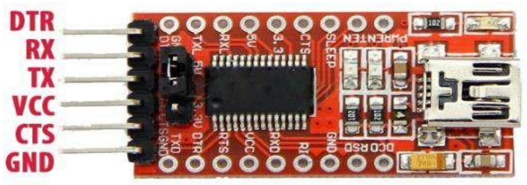


Figure 3: FTDI Module

The ESP32 CAM module is built around the ESP32-S microcontroller chip, which boasts dual-core 32-bit LX6 microprocessors. It integrates a range of connectivity options, including Wi-Fi 802.11 b/g/n (2.4 GHz) with WPA/WPA2 PSK and 802.1X authentication, Bluetooth v4.2 BR/EDR, and BLE. The module incorporates the OV2640 camera module, delivering a resolution of 1600x1200 (2 MP). It is equipped with 4 MB of flash memory and 520 KB of SRAM, while also featuring a MicroSD card slot supporting up to 4 GB of storage. The ESP32 CAM provides multiple interfaces such as UART, SPI, I2C, I2S, PWM, ADC, DAC, and GPIO. Power can be supplied via a 5 V DC input pin or through a USB connection. On-chip sensors, including a temperature sensor and Hall sensor, are integrated. The module's frequency can be adjusted within the range of 80 MHz to 240 MHz

It's important to note that the ESP32 module lacks a built-in USB port, which means that direct program uploads from the Arduino software are not possible. To upload code to the ESP32 module, a separate FTDI driver is required. A UART board, specifically an FTDI USB to TTL serial converter module, is employed for TTL serial communication. This UART board facilitates communication between the computer and the ESP32 module, enabling program uploads and data exchange.

The ESP32 module requires a voltage supply of either 3.3 V or 5 V DC. In order to establish communication with microcontroller development boards such as ESP and Arduino, which do not have built-in USB interfaces, breakout boards with FTDI FT232R chips and USB interfaces are commonly utilized. These breakout boards offer convenient transmit/receive (Tx/Rx) and breakout points for various serial applications. Acting as converters, they enable seamless communication between the computer and microcontroller development boards, facilitating data exchange and program uploads.

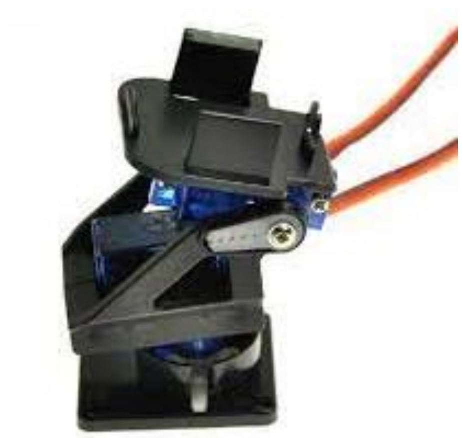


Figure 4: Pan/Tilt Bracket

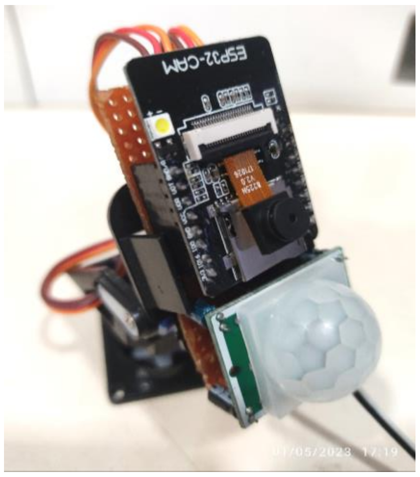


Figure 5: ESP32 Cam with PIR Sensor Assembled Module

Figure 4 shows Pan and Tilt bracket for setting up and controlling two servo motors and interfacing it with an ESP32 to access the camera's video stream from another device.

Figure 5 shows the complete hardware controller section, in this mode, the power adapter should be connected so that the system captures videos continuously and sends the video towards the software / monitor system for further image processing.

Since there is no USB port in this ESP32 module, one cannot directly upload programs to it. So, in order to upload code from Arduino software to this ESP32 module, we required a FTDI USB to TTL serial converter.

Software Implementation

▪ Arduino

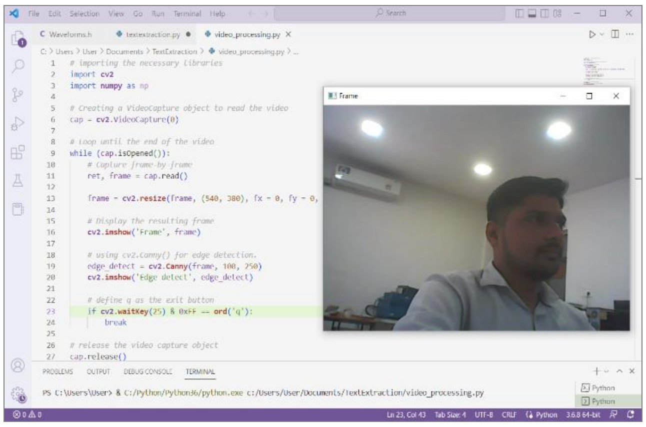
Arduino is a company that specializes in open-source hardware and software, offering a platform for designing and manufacturing microcontrollers and kits used in creating digital devices. The Arduino software is distributed under either the GNU General Public License (GPL) or the GNU Lesser General Public License (LGPL). Arduino boards employ a variety of controllers and microprocessors. These boards feature a combination of digital and analog input/output (I/O) pins, enabling them to interface with expansion boards, controllers, sensors, and modules for prototyping and circuit development. Many of these boards also incorporate serial communication interfaces, and several models support Universal Serial Bus (USB) connectivity for program uploads to the controller. Microcontrollers that are compatible with Arduino can be programmed using the embedded C language, utilizing a standardized API inspired by the Processing language. The programming is facilitated through a modified version of the Processing Integrated Development Environment (IDE), referred to as the Arduino IDE [10].

▪ Python

Python is a highly versatile programming language renowned for its applications in web development (server-side), software development, mathematics, and system scripting. In our system, Python played a crucial role in GUI design, image processing, and the creation of executable files specifically for Windows operating systems, primarily for software development purposes. The Tkinter library was utilized for GUI design, while OpenCV and TensorFlow were employed for face detection and recognition. Tkinter, also known as the tkinter package, is a standard Python interface GUI toolkit that is compatible with various platforms, including Unix, Windows, and MacOS systems. OpenCV, an acronym for Open Source Computer Vision Library, is a free and open-source software library that facilitates machine learning and artificial intelligence tasks, serving as a comprehensive infrastructure for integrating machine perception into computer vision applications and commercial products.

To establish a connection with the camera module in our executable file, the IP address of the camera node is required. This IP address can be assigned as a static IP through Arduino code or obtained dynamically using the Arduino serial monitor to identify the allocated IP address from the network. In our Python code, the obtained IP address is utilized to access live streaming, which is facilitated through the 'urllib' library.

For generating voice alerts based on image recognition, we utilize pre-stored messages in the program and convert text to speech using the 'pyttsx' library. To send email notifications to the owners' personal devices, a Python mail server is employed, utilizing the 'smtplib' library specifically for sending emails. In case of an intruder detection, the camera image is also attached to the email.


File Edit Selection View Go Run Terminal Help
Waveforms.h
textextraction.py
video_processing.py
C: > Users > User > Documents > TextExtraction > video_processing.py > ...
Frame
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
# importing the necessary libraries
import cv2
import numpy as np

# Creating a VideoCapture object to read the video
cap = cv2.VideoCapture(0)

# Loop until the end of the video
while (cap.isOpened()):
    # Capture frame-by-frame
    ret, frame = cap.read()

    frame = cv2.resize(frame, (540, 380), fx = 0, fy = 0,

    # Display the resulting frame
    cv2.imshow('Frame', frame)

    # using cv2.Canny() for edge detection.
    edge_detect = cv2.Canny(frame, 100, 250)
    cv2.imshow('Edge detect', edge_detect)

    # define q as the exit button
    if cv2.waitKey(25) & 0xFF == ord('q'):
        break

# release the video capture object
cap.release()
```

```
PS C:\Users\User> & C:/Python/Python36/python.exe c:/Users/User/Documents/TextExtraction/video_processing.py
```

Figure 6: Desktop of VS Code for Python Coding

It's important to note that the aforementioned statements are based on the general functionality and usage of the mentioned libraries and modules. Actual implementation may require additional considerations and specific code configurations.

TensorFlow, developed by the Google Brain team, is an open-source software library widely used for research and production purposes. It finds applications in various domains, including the medical field, social media, search engines, education, and the retail industry. These examples merely represent potential applications of TensorFlow, as its usage extends beyond the mentioned scenarios. TensorFlow provides extensive capabilities for machine learning and artificial intelligence tasks [11].

At the end of the developed executable file, we are accessing images continuously from the ESP32 camera module, when the system responds to motion detected in the input detected data. Below figures 7 show the desktop of Graphical User Interface designed for monitoring.

- Executable File for Windows OS

We utilized the 'auto-py-to-exe' tool to generate a standalone executable file for our GUI-based Python application. This application was developed using Tkinter and various other libraries. The executable file can be installed and executed on any Microsoft Windows operating system without the need for a separate Python installation. For Mac and Linux users, the PyInstaller command line tools can be used to create the application with specific instructions. By specifying relevant arguments, we can include packaged libraries, icons, and other necessary components.

In the accompanying figure, we present the basic face identification algorithms employed in our system. The figure also showcases a GUI window created using the Tkinter library, providing camera access and incorporating face recognition. This window includes functionality for calling and playing specific voice instructions or alerts from its internal database.

Following figure shows a GUI window designed in Tkinter library and camera access with face recognition. Also, this window calls and play a particular voice instruction or alert from its database.

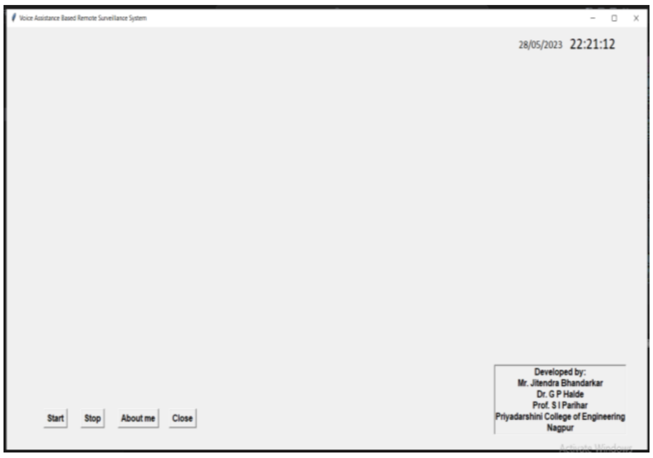


Figure 7: GUI/Executable file for Windows OS

System Architecture

A surveillance system is an efficient solution for monitoring and detecting suspicious activities. Traditional systems are well-known for their cost-effectiveness and operational efficiency. However, the continuous power consumption of these systems can be significant. To address this issue, connectivity options are employed to enable real-time data transfer. Long-range data transfer is achieved by utilizing internet connectivity, specifically through GPRS, ensuring a stable and reliable system suitable for critical conditions. The system operates by detecting motion, activating the camera, capturing images of intruders, performing recognition, and notifying the owner through a monitor screen with voice alerts in case the person is unrecognized.

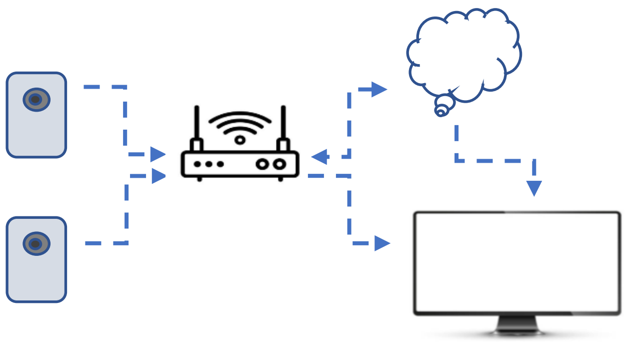


Figure 8: Basic Architecture of Project

In this project, two distinct camera nodes were utilized. Real-time videos from both camera nodes are accessed using Python programming in conjunction with the OpenCV library. By utilizing the IP address and the urllib library, direct access to the cameras can be established over the internet. The system supports the integration of multiple camera units or sensor nodes, whether through wired or wireless connections.

Observations and Results

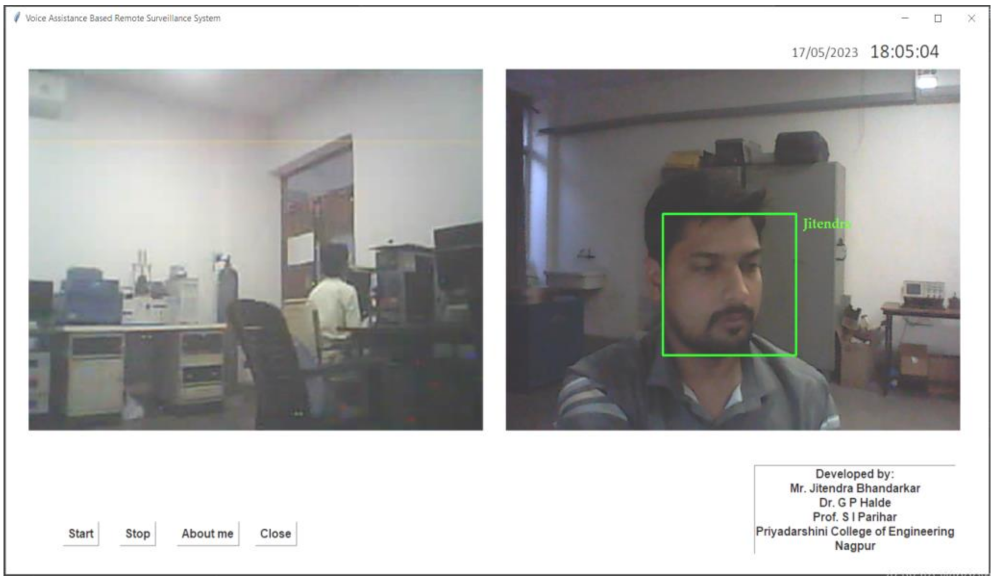
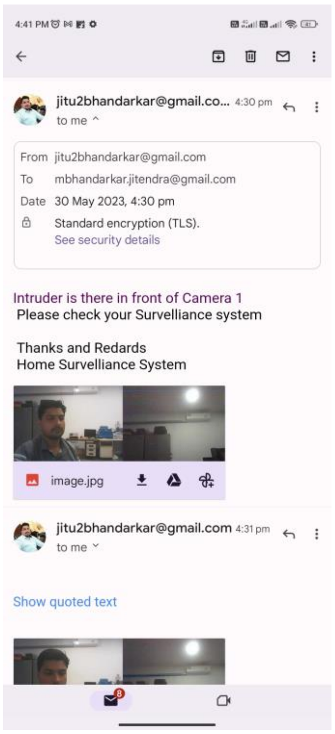


Figure 9: GUI/Executable file for Windows OS

To send emails from the Python IDE using an email address, the smtp module is utilized. However, it is important to note that using the email address and password directly as a sender in Python programming may not be sufficient for all email providers. Some providers may require additional configurations or the use of app-specific passwords. Therefore, it is necessary to ensure that the required permissions and configurations are in place to successfully send emails via SMTP. For example, in the case of Gmail, generating an app password from the Gmail security settings may be necessary.

To distribute the system, an executable file (.exe) has been included alongside the thesis. This file can be stored on a portable storage device and installed on a Windows operating system. When executed, the file plays a welcome message informing the user about the internet connectivity with the Windows system. The received mail is displayed in Figure 11.


4:41 PM

jitu2bhandarkar@gmail.co... 4:30 pm
to me

From jitu2bhandarkar@gmail.com
To mbhandarkar.jitendra@gmail.com
Date 30 May 2023, 4:30 pm
Standard encryption (TLS).
See security details

Intruder is there in front of Camera 1
Please check your Surveillance system

Thanks and Redards
Home Surveillance System

image.jpg

jitu2bhandarkar@gmail.com 4:31 pm
to me

Show quoted text

Figure 10: Email Alert from Sender Side

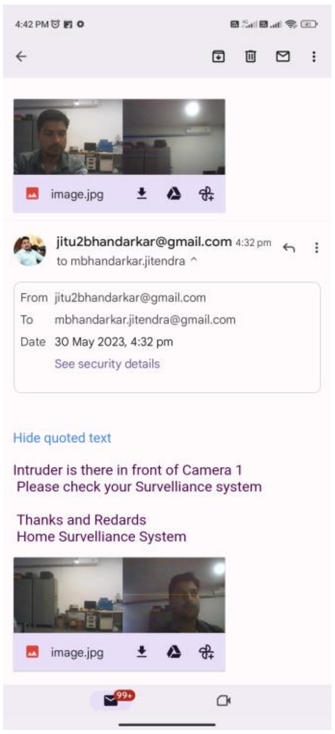

4:42 PM

image.jpg

jitu2bhandarkar@gmail.com 4:32 pm
to mbhandarkar.jitendra

From jitu2bhandarkar@gmail.com
To mbhandarkar.jitendra@gmail.com
Date 30 May 2023, 4:32 pm
See security details

Hide quoted text

Intruder is there in front of Camera 1
Please check your Surveillance system

Thanks and Redards
Home Surveillance System

image.jpg

Figure 11: Email Alert at Receiver Side

Furthermore, when an email is sent to the recipient's email address, a corresponding voice alert is generated based on image recognition and played in the background of the software.

Conclusion

Under this project, we had created a voice assistant which is capable of running and playing all commands from its database according the image recognition. We are getting intruder images at our mail ID properly. Here we are accessing two streaming’s from different two independent camera modules. We are even trying it to make it highly user friendly by implementing the GUI for our particular VA so that it helps them more to understand it in a better manner and finally it runs and working properly.

References

[1]Ruiguan Ge, Zhenfang Shan, and Hao Kou, “An Intelligent Surveillance System Based on Motion Detection,” IEEETransl. China, pp. 306-309, 2011.
[2]Saroja Kanta Panda and Sushanta Kumar Sahu, “Design of IoT-Based Real-Time Video Surveillance System Using Raspberry Pi and Sensor Network,” Springer. vol. 185, pp. 115-124, 2021.
[3]Qinsheng Du, and Jiling Tang, “Design of the ARM Based Remote Surveillance System,” IEEE Transl. China,pp. 336-338, 2012.
[4]Arvin Joseph Kumar Jayakumar, and Muthulakshmi, “Raspberry Pi-Based Surveillance System with IoT,” Springer, vol. 2, pp. 173–185, 2018.

[5]Rohan Namdeo, Sahil Sharma, Varun Anand, Chanchal Lohi, “Smart Automated Surveillance Syatem using Raspberry pi” International Journal of Recent Technology and Engineering vol. 9, July 2020.
[6]Sruthy. S, S. Yamuna, and Sudhish N. George, Member, “An IoT based Active Building Surveillance System using Raspberry Pi and NodeMCU” IEEE, 2017.
[7]Chinmaya Kaundanya, Omkar Pathak, Akash Nalawade, Sanket Parode “Smart Surveillance System using Raspberry Pi and Face Recognition”, IJARCCE,vol. 6, issue 4, April 2017.
[8]Singoee Sylvestre Sheshai “Raspberry pi Based Security System”, University of Nairobi Department of Electrical and Information Engineering, May 2016.
[9]Miss. Mahima F. Chauhan, Prof. Gharge Anuradha P. “Real Time Video Surveillance System Based on Embedded Web Server Raspberry Pi B+ Board”, National Conference On Recent Research In Engineering And Technology, E-ISSN: 2348-4470, 2015.
[10]https://www.espressif.com/en/news/ESP32_CAM.
[11]https://www.python.org/about/gettingstarted/