# DAYANANDA SAGAR UNIVERSITY

SCHOOL OF ENGINEERING

Department of Computer Science & Engineering (AIML)

Devara Kaggalahalli, Harohalli Kanakapura Road,

Ramanagara - 562112

Karnataka, India

**A**

**Special Topic-1**
**Report**
**On**

## "INTELLIGENT SURVEILLANCE SYSTEM"

## Department of Computer Science Engineering (AI & ML)

**SUBMITTED BY**

**Chethan S** **ENG22AM0084**

**Gudla Monish** **ENG22AM0096**

**Under the supervision of**
**Dr. Vegi Fernando**

**Designation**
**Dept. of AIML, SOE, DSU**

**2023 - 2024**

# DAYANANDA SAGAR UNIVERSITY

School of Engineering
Department of Computer Science & Engineering (AIML)
Devara Kaggalahalli, Harohalli, Kanakapura Road,
Ramanagara – 562112
Karnataka, India

## Department of Computer Science Engineering (AI & ML)

# CERTIFICATE

This is to certify that the Special Topic-1(22AM2406) project work titled **"INTELLIGENT SURVEILLANCE SYSTEM"** is carried out by **Chethan S ENG22AM0084, Gudla Monish ENG22AM0096,** bonafide students of Bachelor of Technology in Computer Science and Engineering (AI&ML) at the School of Engineering, Dayananda Sagar University in partial fulfillment for the award of degree in Bachelor of Technology in Computer Science and Engineering (AI&ML), during the year **2023-2024**.

---
Signature of Guide

**Dr. Vegi Fernando**
**Associate Professor**
**Dept. of AIML, SOE, DSU**

---
Signature of Chairperson

**Dr. Jayavrinda Vrindavanam, Ph.D**
**Professor and Chairperson,**
**Dept. of AIML, SOE, DSU**

# DECLARATION

We, **Chethan S ENG22AM0084, and Gudla Monish ENG22AM0096** are students of the fourth semester B.Tech in **Computer Science and Engineering(AI&ML)**, at the School of Engineering, **Dayananda Sagar University**, hereby declare that the Special Topic-1 titled **"INTELLIGENT SURVEILLANCE SYSTEM"** has been carried out by us and submitted in partial fulfillment for the award of degree in **Bachelor of Technology in Computer Science and Engineering(AI&ML)** during the academic year **2023-2024**.

# ACKNOWLEDGEMENT

It is a great pleasure for us to acknowledge the assistance and support of many individuals who have been responsible for the successful completion of this project work.

First, we take this opportunity to express our sincere gratitude to the School of Engineering & Technology, Dayananda Sagar University for providing us with a great opportunity to pursue our Bachelor's degree in this institution.

We would like to thank **Dr. Udaya Kumar Reddy K R.**, Dean, **School of Engineering & Technology**, **Dayananda Sagar University** for his constant encouragement and expert advice. It is a matter of immense pleasure to express our sincere thanks to **Dr. Jayavrinda Vrindavanam, Department Chairperson**, **Computer Science and Engineering (AI&ML)**, **Dayananda Sagar University,** for providing the right academic guidance that made our task possible.

We would like to thank our guide **Associate Professor Vegi Fernando**, **Dept. of Computer Science and Engineering(AI&ML)**, **Dayananda Sagar University**, for sparing his/her valuable time to extend help in every step of our Special Topic-1 work, which paved the way for smooth progress and the fruitful culmination of the research.

We would like to thank our **Special Topic-1 Coordinators Dr. Jayavrinda Vrindavanam, Professor, Dr. Joshuva Arockia Dhanraj, Associate Professor, and Dr. Mude Nagarjuna Naik, Assistant Professor** and all the staff members of Computer Science and Engineering (AI&ML) for their support.

We are also grateful to our family and friends who provided us with every requirement throughout the course. We would like to thank one and all who directly or indirectly helped us in the Special Topic-1 work.

# TABLE OF CONTENTS

# LIST OF ABBREVIATIONS

| | |
|---|---|
| ISS | Intelligent surveillence system |
| OpenCV | Open computer vision |
| HC | haar classifiers |
| | |
| | |

# LIST OF FIGURES

# LIST OF TABLES

# ABSTRACT

This project presents the design and implementation of a smart camera system that records video exclusively upon detecting human movement. The primary objective is to optimize storage and energy usage by ensuring that recording occurs only when necessary. The system utilizes a motion detection algorithm, which processes input from a camera sensor to identify the presence of a person. Upon detection, the camera initiates video recording. When no movement is detected, the recording ceases, thereby conserving storage space and extending the operational lifespan of the system.

The core components of the project include a camera module, a motion detection sensor, a microcontroller for processing, and storage for video files. Motion detection is achieved through computer vision techniques, which accurately distinguish human movement from other types of motion, reducing false positives and enhancing the system's reliability.

Key features of the smart camera system include real-time processing, efficient video storage management, and the capability to integrate with existing security frameworks. The system is designed to be easily deployable in various environments such as homes, offices, and public spaces, providing an intelligent surveillance solution that balances security needs with resource efficiency.

This project demonstrates a practical application of motion detection in surveillance, highlighting the benefits of adaptive recording in terms of storage optimization and energy conservation. Future improvements could involve refining the motion detection algorithm, incorporating advanced AI techniques for better accuracy, and extending the system's functionality to include remote monitoring and alerts.

# CHAPTER 1

# INTRODUCTION

# CHAPTER 1 INTRODUCTION

Surveillance systems are pivotal for maintaining security across various domains, including residential, commercial, and public environments. However, traditional surveillance approaches often suffer from inefficiencies such as continuous recording, leading to storage overload and unnecessary resource consumption. To address these challenges, this project introduces an Intelligent Surveillance System (ISS) designed to record video selectively, triggered solely by human motion detection.

## 1.1 Background

Reason why *surveillance systems* are present everywhere underscores their importance in modern security protocols. though effective, are marred by indiscriminate recording practices, resulting in information overload and resource wastage.

### 1.1.1 Objective

The primary aim of this project is to develop an intelligent surveillance system that optimizes resource usage by recording video only in response to human motion. By leveraging motion detection technology, the system will enhance efficiency, reduce storage requirements, and minimize energy consumption.

#### 1.1.1.1 Significance

The significance of this project lies in its potential to revolutionize surveillance practices, making them more efficient, and sustainable. By intelligently recording video based on human activity(motion.

## 1.2 SCOPE

The social impact would be the increased awareness of security and Empowerment through technology and the Environmental impact would be reduced energy consumption.

# 1.3 FIGURES

Including Sample images for the following techniques used in our project:

**Thresholding:** apply a threshold to the difference image obtained from the previous step like capturing video frames, pre-processing, and video modeling.This helps to filter out small changes and highlight significant movements.

**Contour detection:** The next step is to detect contours in the thresholded image using OpenCV's contour detection function.

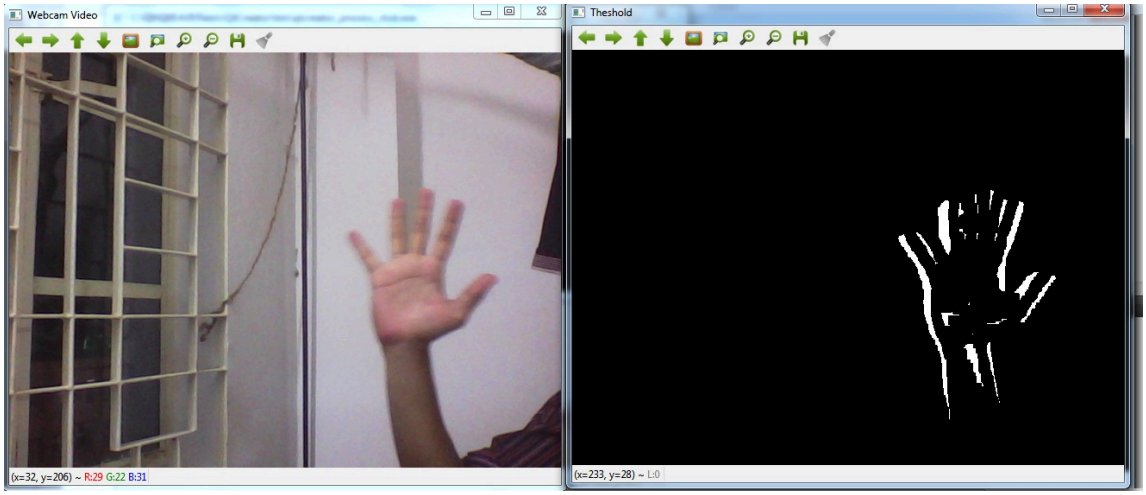


*Figure 1.1 Thresholding*

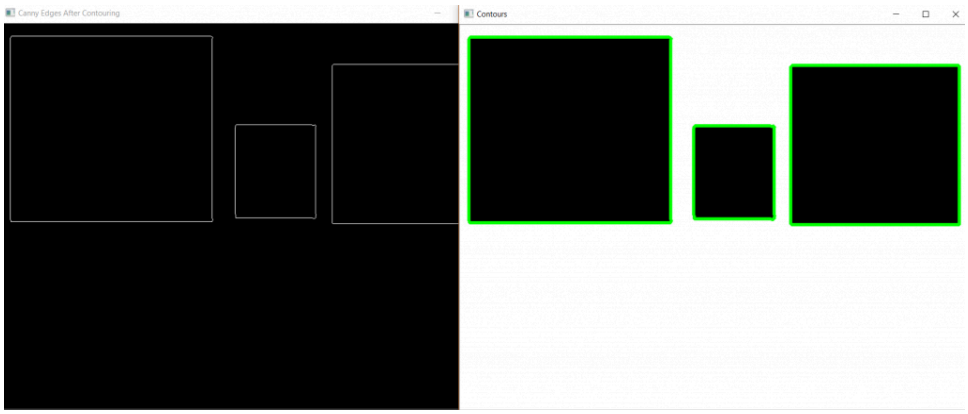


*Figure 1.2: Contour detection example*

# CHAPTER 2
# PROBLEM DEFINITION

# CHAPTER 2 PROBLEM DEFINITION

Traditional surveillance systems suffer from inefficiencies due to continuous recording practices, leading to excessive storage usage and wasteful resource consumption. This results in challenges such as high maintenance costs, limited storage capacity, and difficulties in managing vast amounts of irrelevant footage. Furthermore, the indiscriminate recording approach often fails to provide timely alerts or relevant information during security incidents, compromising the effectiveness of the surveillance system. To address these challenges, there is a need for an Intelligent Surveillance System (ISS) capable of selectively recording video based on human motion detection. The ISS should efficiently utilize resources by activating recording only when significant motion indicative of human presence is detected, thus minimizing storage requirements and conserving energy. Additionally, the system should be able to distinguish between human activity and other forms of motion, such as animals or environmental factors, to reduce false alarms and ensure accurate detection. The primary objective of this project is to design, develop, and implement an ISS that enhances the efficiency, effectiveness, and sustainability of surveillance operations. By intelligently capturing and analyzing video footage in real-time, the ISS aims to improve security monitoring, facilitate rapid response to security incidents, and optimize resource utilization in surveillance environments.

# CHAPTER 3

# LITERATURE REVIEW

# CHAPTER 3 LITERATURE REVIEW

Surveillance systems play a critical role in modern security protocols, spanning from residential protection to public safety. The integration of computer vision and image processing technologies has revolutionized the capabilities of surveillance systems, enabling more intelligent solutions. In this literature review, we wish to explain and explore works that contribute to the development of intelligent surveillance systems.

*Bradski and Kaehler* in their book "*Learning OpenCV*" provide foundational knowledge on computer vision techniques and their applications. They lay the groundwork for understanding the principles behind image processing algorithms, including motion detection and object recognition. The book served as a fundamental resource for us to leverage OpenCV for surveillance system development.

*S. Suresh, J. Bhavya, S. Sakshi, K. Varun, and G. Debarshi* present a research paper titled "*Home Monitoring and Security System*," where they propose a comprehensive home surveillance solution. Their system integrates various sensors and cameras to monitor indoor and outdoor environments, with a focus on remote monitoring and alert functionalities.

*T. Prathaban, W. Thean, and M.I.S. Mohd Sazali* introduce a *vision-based home security system utilizing OpenCV on Raspberry Pi 3.* Their research demonstrates the feasibility of deploying computer vision algorithms on embedded platforms for real-time surveillance applications. By leveraging the capabilities of Raspberry Pi and OpenCV, our project was inspired, although we modified it to work with just a camera module and a computer system.

The reviewed literature highlights advancements in intelligent surveillance, emphasizing algorithm integration, hardware utilization, and remote monitoring. It underscores efficient resource usage and real-time response for effective surveillance.

This research aims to advance intelligent surveillance by computer vision techniques. It targets enhanced security, resource optimization, and real-time response, contributing to modern security infrastructure evolution.

# CHAPTER 4
# PROJECT DESCRIPTION

# CHAPTER 4 PROJECT DESCRIPTION

Our project aims to develop an intelligent surveillance system utilizing computer vision techniques, particularly Haar classifiers. These classifiers will enable real-time human detection within surveillance footage, triggering selective video recording based on detected motion.

The system will consist of a camera module for video capture, integrated with Haar cascades trained for face and body detection. Upon detecting significant motion indicative of human presence, the system will activate video recording, optimizing storage usage by capturing only relevant footage.

Key features include real-time motion detection facilitated by Haar classifiers, ensuring efficient identification of human subjects in surveillance streams. This selective video recording capability enhances efficiency by activating recording only when significant human motion is detected, minimizing storage requirements and reducing unnecessary data accumulation.

Furthermore, the system will be designed for seamless integration with existing surveillance infrastructure, ensuring compatibility with various hardware platforms and enabling deployment in diverse surveillance environments.

Resource optimization is a core aspect, with a focus on efficient utilization of computational resources for real-time processing and response. By leveraging Haar classifiers, we aim to develop a robust and efficient solution for security monitoring, addressing the evolving challenges in modern security infrastructure.

# CHAPTER 5
# METHODOLOGY

# CHAPTER 5 METHODOLOGY

Our methodology encompasses a multifaceted approach to developing an Intelligent Surveillance System (ISS) with a focus on efficient motion detection and selective video recording. Both established techniques and innovative strategies are utilized to achieve objectives effectively.

**1. Utilization of Haar Cascades:** Haar cascades, a well-established technique in computer vision, are integrated for real-time face and body detection. Robustness and efficiency in detecting predefined patterns make them ideal for identifying human subjects within surveillance footage.

**2. Motion Detection Algorithms:** Advanced motion detection algorithms are employed to analyze consecutive video frames and identify areas of significant change. Techniques such as absolute difference calculation, Gaussian blur, and contour detection enable precise localization of motion, facilitating accurate triggering of video recording.

**3. Selective Video Recording Mechanism:** A selective video recording mechanism is implemented. Upon detecting significant motion indicative of human presence, video recording is activated to capture relevant footage. This strategy minimizes storage requirements and reduces the need for manual video analysis.

**4. Real-time Processing and Response:** The ISS prioritizes real-time processing and response capabilities to ensure timely detection and recording of security incidents. Optimized algorithms and parallel processing techniques are leveraged to minimize latency and enhance responsiveness to motion events.

**5. System Integration and Compatibility:** Seamless integration with existing surveillance infrastructure and compatibility with diverse hardware platforms are emphasized. This approach enables easy deployment of the ISS in various surveillance environments, ranging from homes to commercial establishments.

**6. Iterative Development and Optimization:** An iterative development approach is adopted, allowing for continuous refinement and optimization of the ISS. Through iterative testing and feedback loops, detection accuracy, false positives, and overall system performance are improved.

**7. Error Handling and Robustness:** Robust error handling mechanisms are incorporated to ensure system stability and reliability. Strategies for handling empty

images, contour detection failures, and resource management are implemented to mitigate potential issues and ensure uninterrupted operation.

**8. Documentation and Knowledge Sharing:** Throughout the project lifecycle, comprehensive documentation and knowledge-sharing practices are followed. Detailed documentation of methodologies, algorithms, and implementation details enable effective collaboration and facilitate future enhancements or modifications to the ISS.

By adopting these strategies and techniques, a robust, efficient, and scalable Intelligent Surveillance System capable of enhancing security monitoring and resource utilization in diverse surveillance environments is developed.

## Methods of data collection

Our method of data collection for the Intelligent Surveillance System (ISS) project primarily relies on real-time video streams captured by the OpenCV library using a webcam or other video input device. The video feed serves as the primary source of data for our analysis, enabling us to detect and analyze motion events for selective video recording. We did not engage in traditional case study methods involving texts or images; instead, our analysis is based on the real-time video data captured by the ISS during its operation. This data includes frames from the video feed processed using computer vision techniques, such as motion detection with Haar cascades. Since our data collection process involves real-time video capture and processing, there was no need for manual selection or preparation of data. However, before analyzing the video frames for motion detection, the OpenCV library handles basic preprocessing steps such as converting images to grayscale, applying Gaussian blur, and thresholding to enhance the quality of the data and facilitate accurate motion detection. Overall, our data collection methodology is centered around real-time video capture and processing using OpenCV, with no manual intervention required for data selection or preparation

# CHAPTER 6
# RESULTS AND ANALYSIS

# CHAPTER 6 RESULTS AND ANALYSIS

## Implemented code in Pycharm(IDE)

main.py

```python
import cv2
import time
import datetime

cam = cv2.VideoCapture(0)

face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + "haarcascade_frontalface_default.xml")
body_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + "haarcascade_fullbody.xml")

detection = False
detection_stopped_time = None
timer_started = False
SECONDS_TO_RECORD_AFTER_DETECTION = 5

frame_size = (int(cam.get(3)), int(cam.get(4)))
fourcc = cv2.VideoWriter_fourcc(*"mp4v")

while cam.isOpened():
    ret, frame1 = cam.read()
    ret, frame2 = cam.read()

    diff = cv2.absdiff(frame1, frame2)

    if diff is not None:
        gray = cv2.cvtColor(diff, cv2.COLOR_RGB2GRAY)
        if gray is not None:
            blur = cv2.GaussianBlur(gray, ksize: (5, 5), sigmaX: 0)
            _, thresh = cv2.threshold(blur, thresh: 20, maxval: 255, cv2.THRESH_BINARY)
            dilated = cv2.dilate(thresh, kernel: None, iterations=3)
            contours, _ = cv2.findContours(dilated, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)

            if any(cv2.contourArea(c) > 5000 for c in contours):
                if not detection:
                    detection = True
                    current_time = datetime.datetime.now().strftime("=Special_Topic-%d-%m-%Y-%H-%M-%S")
                    out = cv2.VideoWriter(f"{current_time}.mp4", fourcc, 20, frame_size)
                    print("Started Recording!")

            elif detection:
                if timer_started:
                    if time.time() - detection_stopped_time >= SECONDS_TO_RECORD_AFTER_DETECTION:
                        detection = False
                        timer_started = False
                        out.release()
                        print('Stop Recording!')
                else:
                    timer_started = True
                    detection_stopped_time = time.time()

            if detection:
                out.write(frame1)

            for c in contours:
                if cv2.contourArea(c) < 5000:
                    continue
                x, y, w, h = cv2.boundingRect(c)
                cv2.rectangle(frame1, (x, y), (x + w, y + h), (0, 255, 0), 2)

            cv2.imshow( winname: 'Special_Topic', frame1)

            if cv2.waitKey(10) == ord('q'):
                break
```

```
        else:
            print("Error: Gray image is empty")
    else:
        print("Error: Difference image is empty")

out.release()
cam.release()
cv2.destroyAllWindows()

```

*Figure 6.1: code block in pycharm*

## Output in terminal

```
/Users/chethans/PycharmProjects/movement_also/venv/bin/python /Users/chethans/PycharmProjects/movement_also/main.py
Started Recording!
Stop Recording!
Started Recording!
Stop Recording!
```

*Figure 6.2: output in terminal*

## Output (video automatically being saved in the mentioned location, with timestamp)

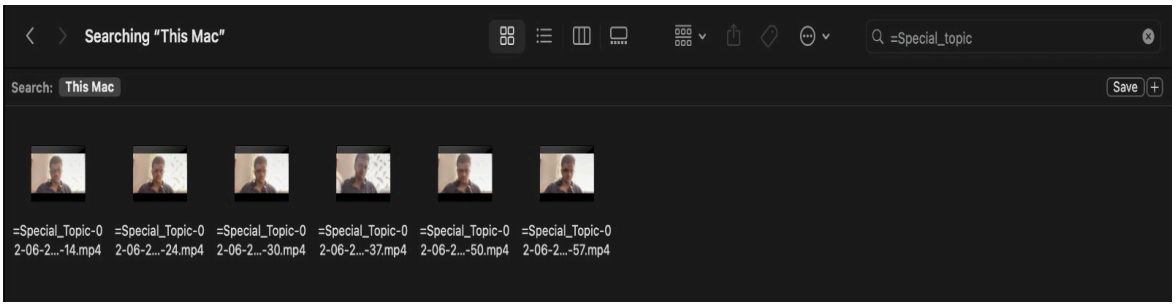


*Figure 6.3: saved video directory*

## Test case1(when object is near)

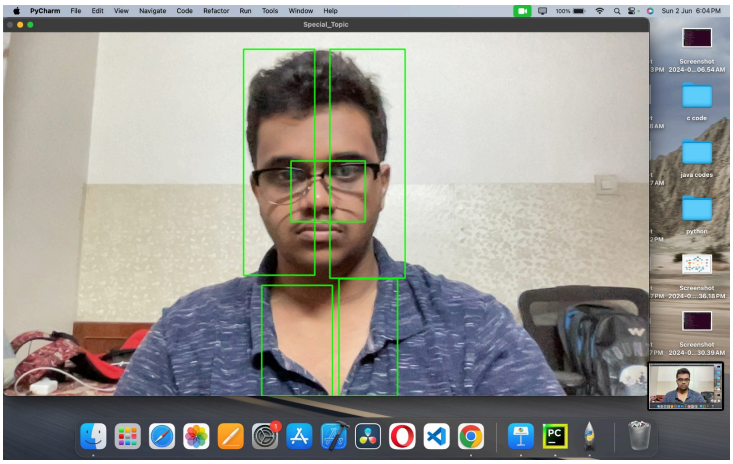

*Figure 6.4: (above) when person in motion*

### Test case 2 (when the object is in motion)

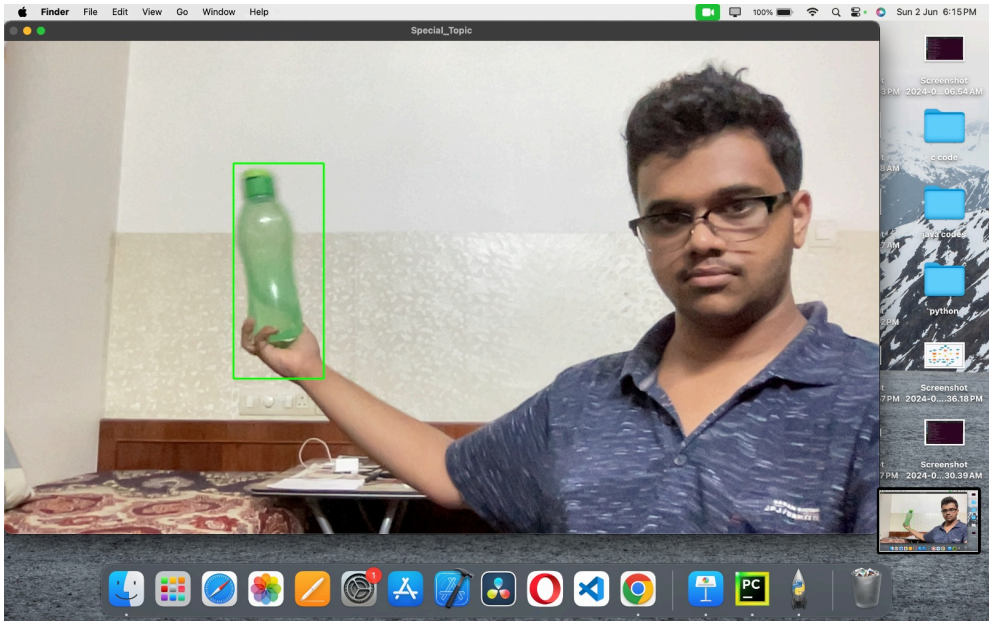


*Figure 6.5 when object in motion:*

### Evaluation of results in the form of tables

| Test Cases | No.of Times it recognized motion | No.of Times it did not recognize motion |
|---|---|---|
| When an Object in motion near the camera | 47 of 50 | 3 of 50 |
| When an Object in motion far from the camera | 45 of 50 | 5 of 50 |

Therefore, the accuracy of the ISS for detecting motion near the camera is **94%**, and for detecting motion far from the camera is **90%**

# CONCLUSION AND FUTURE WORK

In conclusion, our project has successfully developed an *Intelligent Surveillance System (ISS)* leveraging computer vision techniques, particularly Haar cascades, for efficient motion detection and selective video recording. The *ISS* demonstrates robustness and efficiency in detecting human subjects within surveillance footage, optimizing storage usage by activating video recording only when significant motion is detected. The methodology employed in this project has enabled the integration of advanced motion detection algorithms and real-time processing capabilities, resulting in a scalable and adaptable surveillance solution suitable for various environments. Moving forward, the ISS holds promise for enhancing security monitoring and resource utilization in both residential and commercial settings.

Looking ahead, we would like to explore more *OpenCV* projects to expand our understanding and expertise in computer vision applications. Additionally, we are interested in integrating the ISS with embedded systems like Arduino or STM to enhance its versatility and applicability in different scenarios. Furthermore, there is potential to develop web-based interfaces for remote monitoring and control of the ISS, providing users with convenient access to surveillance footage and system settings. These future endeavors aim to further enhance the functionality and usability of the ISS, making it an even more valuable tool for security monitoring and management.

# REFERENCES

[1] G. Bradski and A. Kaehler, "Learning OpenCV," O'Reilly Media, 2008. [Online]. Available: https://www.oreilly.com/library/view/learning-opencv/9780596516130/

[2] S. Suresh, J. Bhavya, S. Sakshi, K. Varun, and G. Debarshi, "Home Monitoring and Security System," IEEE, Apr. 6, 2017. [Online]. Available: https://ieeexplore.ieee.org/document/7892665, DOI: 10.1109/ICTBIG.2016.7892665

[3] T. Prathaban, W. Thean, and M.I.S. Mohd Sazali, "A Vision-Based Home Security System Using OpenCV on Raspberry Pi 3," AIP Publishing, Nov. 11, 2019: https://pubs.aip.org/aip/acp/article/2173/1/020013/746630/A-vision-based-home-security-system-using-OpenCV DOI: [10.1063/1.5133928]

# APPENDIX - A

**CODE**

```
import cv2

import time

import datetime

cam = cv2.VideoCapture(0)

# Load Haar cascades for face and body detection

face_cascade=cv2.CascadeClassifier(cv2.data.haarcascades+"haarcascade_frontalface_default.xml")

body_cascade=cv2.CascadeClassifier(cv2.data.haarcascades+"haarcascade_fullbody.xml")

detection = False

detection_stopped_time = None

timer_started = False

SECONDS_TO_RECORD_AFTER_DETECTION = 5

frame_size = (int(cam.get(3)), int(cam.get(4)))

fourcc = cv2.VideoWriter_fourcc(*"mp4v")

while cam.isOpened():

    ret, frame1 = cam.read()

    ret, frame2 = cam.read()

    # Calculate absolute difference between consecutive frames

    diff = cv2.absdiff(frame1, frame2)

    if diff is not None:

        gray = cv2.cvtColor(diff, cv2.COLOR_RGB2GRAY)
```

```
            if gray is not None:
                blur = cv2.GaussianBlur(gray, (5, 5), 0)
                _, thresh = cv2.threshold(blur, 20, 255, cv2.THRESH_BINARY)
                dilated = cv2.dilate(thresh, None, iterations=3)
 contours,_=cv2.findContours(dilated,cv2.RETR_TREE,
cv2.CHAIN_APPROX_SIMPLE)
                    if any(cv2.contourArea(c) > 5000 for c in contours):
                  if not detection:
                     detection = True
                                                                  current_time =
    datetime.datetime.now().strftime("=Special_Topic-%d-%m-%Y-%H-%M-%S")
                     out = cv2.VideoWriter(f"{current_time}.mp4", fourcc, 20, frame_size)
                     print("Started Recording!")

                # Stop recording after a specified time period
                elif detection:
                   if timer_started:
                                          if time.time() - detection_stopped_time >=
    SECONDS_TO_RECORD_AFTER_DETECTION:
                        detection = False
                        timer_started = False
                        out.release()
                        print('Stop Recording!')
                   else:
                      timer_started = True
```

```
                detection_stopped_time = time.time()
              if detection:
            out.write(frame1)
                for c in contours:
            if cv2.contourArea(c) < 5000:
                continue
            x, y, w, h = cv2.boundingRect(c)
            cv2.rectangle(frame1, (x, y), (x + w, y + h), (0, 255, 0), 2)
        # Display processed frame
        cv2.imshow('Special_Topic', frame1)
        # Break the loop if 'q' is pressed
        if cv2.waitKey(10) == ord('q'):
            break
    else:
        print("Error: Gray image is empty")
else:
    print("Error: Difference image is empty")

# Release resources
out.release()
cam.release()
cv2.destroyAllWindows()
```