A

MINI-PROJECT REPORT

ON

CROWD LIMITATION ALERTING SYSTEM

Submitted in partial fulfillment of the requirements for the award of the degree

of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE ENGINEERING

Submitted By:

M. SRIHARSHA (21UP1A05G3)
B.RAVALI (21UP1A05E1)
P.USHASARVANI (21UP1A05H0)

Under the Guidance of

Mrs. Geetha Bhavani

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
VIGNAN'S INSTITUTE OF MANAGEMENT AND TECHNOLOGY FOR WOMEN
(An Autonomous Institution)
(Affiliated to Jawaharlal Nehru Technological University Hyderabad, Accredited by NBA, NAAC with A+)
Kondapur(Village), Ghatkesar (Mandal), Medchal (Dist.Telangana-501301

(2021-2025)

VIGNAN'S INSTITUTE OF MANAGEMENT AND TECHNOLOGY FOR WOMEN

(An Autonomous Institution)

(Sponsored by Lavu Educational Society)

[Affiliated to JNTUH, Hyderabad & Approved by AICTE, New Delhi]

Kondapur (V), Ghatkesar (M), Medchal –Malkajgiri (D)- 501301.Phone:9652910002/3

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the project work entitled "**CROWD LIMITATION ALERTING SYSTEM**" submitted by **M.Sriharsha(21UP1A05G3), B.Ravali(21UP1A05E1), P.UshaSarvani (21UP1A05H0)** in the partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering**, **Vignan's Institute of Management and Technology for Women** is a record of bonafide work carried by them under my guidance and supervision. The results embodied in this project report have not been submitted to any other University or institute for the award of any degree.

PROJECT GUIDE
Mrs. GeethaBhavani

THE HEAD OF DEPARTMENT
Mrs. M. Parimala
(Associate Professor)

(External Examiner)

VIGNAN'S INSTITUTE OF MANAGEMENT AND TECHNOLOGY FOR WOMEN

(An Autonomous Institution)

(Sponsored by Lavu Educational Society)

[Affiliated to JNTUH, Hyderabad & Approved by AICTE, New Delhi]

Kondapur (V), Ghatkesar (M), Medchal –Malkajgiri (D)- 501301.Phone:9652910002/3

DECLARATION

We hereby declare that the results embodied in the project entitled **"CROWD LIMITATION ALERTING SYSTEM"** is carried out by us during the year 2024-2025 in partial fulfillment of the award of **Bachelor of Technology** in **Computer Science and Engineering** from **Vignan's Institute of Management and Technology for Women** is an authentic record of our work under the guidance of Guide Name. We have not submitted the same to any other institute or university for the award of any other Degree.

M. SrI Harsha (21UP1A05G3)

B. Ravali (21UP1A05E1)

P. Usha Sarvani (21UP5A05H0)

ACKOWLEDGEMENT

We would like to express sincere gratitude to **Dr G. APPARAO NAIDU**, **Principal, Vignan's Institute of Management and Technology for Women** for his timely suggestions which helped us to complete the project in time.

We would also like to thank our madam **Mrs. M. Parimala, Head of the Department and Associate Professor, Computer Science and Engineering** for providing us with constant encouragement and resources which helped us to complete the project in time.

We would also like to thank our Project guide **Mrs.** Geetha Bhavani, **Professor/Associate professor/Assistant Professor, Computer Science and Engineering**, for providing us with constant encouragement and resources which helped us to complete the project in time with his/her valuable suggestions throughout the project. We are indebted to him/her for the opportunity given to work under his/her guidance.

Our sincere thanks to all the teaching and non-teaching staff of Department of Computer Science and Engineering for their support throughout our project work.

M. SrI Harsha (21UP1A05G3)

B. Ravali (21UP1A05E1)

P. Usha Sarvani (21UP5A05H0)

INDEX

LIST OF FIGURES

ABSTRACT

The persistent challenge of overcrowding in public spaces during events and festivals necessitates innovative solutions for efficient crowd management. In this paper, we propose a real-time crowd detection system employing OpenCV and pyttsx3. Our proposed system offers a practical and efficient solution for overcrowding management, providing real-time insights and alerts to facilitate timely interventions in public spaces. The system utilizes Haar cascade classifiers within OpenCV for accurate and rapid identification of pedestrians and upper bodies in video streams. The project initiates by initializing essential components, including a video capturing device, and enters a processing loop where frames are continuously analyzed. The integration of OpenCV and pyttsx3 showcases the synergy between computer vision and audio feedback, contributing to enhanced crowd surveillance and control during various events.

1.INTRODUCTION

1.1 MOTIVATION:

The motivation for developing a real-time crowd detection and overcrowding management system using OpenCV and pyttsx3 stems from the growing need to enhance public safety, improve the efficiency of crowd management, and elevate the user experience in densely populated areas. Overcrowding in public spaces such as transportation hubs, shopping malls, sports venues, and large-scale events poses significant risks, including the potential for stampedes, accidents, and heightened vulnerability during emergencies. Traditional methods of crowd monitoring, which often rely on manual observation and static surveillance systems, are not always capable of responding swiftly or effectively to sudden changes in crowd density. This limitation necessitates the development of a more dynamic and responsive solution.

By leveraging the capabilities of OpenCV for real-time image processing and analysis, this project aims to accurately detect and monitor crowd densities in various environments. OpenCV's robust set of tools and algorithms enables the system to process live video feeds and identify overcrowded areas quickly and efficiently. Integrating pyttsx3, a text-to-speech conversion library, further enhances the system by providing real-time audio alerts. These alerts ensure immediate communication of critical information to authorities or automated response mechanisms, enabling rapid intervention to prevent potential hazards.

The implementation of this system is expected to significantly improve the allocation of resources, allowing for more effective deployment of security personnel and emergency services. Real-time data and alerts facilitate quicker decision-making, ensuring that overcrowded situations are managed promptly to mitigate risks. This proactive approach not only enhances public safety but also contributes to a more pleasant and orderly environment for users, reducing stress and discomfort associated with crowded spaces.

Furthermore, the adaptability of the system to various scales and environments underscores its versatility. Whether in small venues or large public gatherings, the system can be tailored to meet specific needs, making it a valuable tool for diverse applications. By harnessing the power of modern technology, this project addresses a

fundamental challenge in urban management, promoting safer, more efficient, and user-friendly public spaces. The ultimate goal is to create a scalable, real-time solution that can be widely adopted to enhance safety and optimize the use of public spaces in an increasingly crowded world.

1.2 PROBLEM DEFINITION:

The objective of this project is to develop a real-time crowd detection system using OpenCV (Open Source Computer Vision Library) and pyttsx3 (Python Text -to - Speech Library). The system should be capable of accurately detecting and monitoring crowd density in various environments and providing immediate auditory alerts when overcrowding is detected.

Key Components :

1. **Crowd Detection Algorithm**: Develop an algorithm capable of detecting and estimating crowd density in real-time using video feeds.
2. **OpenCV Integration**: Utilize OpenCV for image processing and analysis to implement the crowd detection algorithm.
3. **Auditory Alert System**: Integrate pyttsx3 to provide real-time auditory alerts to authorities or designated personnel when overcrowding is detected.
4. **Real-Time Processing**: Ensure the system processes video feeds in real-time with minimal latency to facilitate timely response to overcrowding situations.
5. **Robustness and Scalability**: Design the system to be robust against variations in lighting, crowd density, and environmental conditions. Ensure scalability to handle different scales of crowds and various types of environments

Objectives :

- Develop a robust crowd detection algorithm capable of accurately estimating crowd density.
- Implement the algorithm using OpenCV to enable real-time processing of video feeds.
- Integrate pyttsx3 to provide immediate auditory alerts when overcrowding is detected.
- Validate the system's accuracy, efficiency, and reliability through extensive testing in different environments.
- Demonstrate the effectiveness of the system in enhancing crowd management and safety measures.

Stake Holders :

- Authorities responsible for crowd management and safety in public spaces.
- Event organizers.
- Transportation agencies.
- Security personnel.

Expected Impact :

- Improved safety measures in crowded public spaces.
- Enhanced efficiency in crowd management operations.
- Timely response to overcrowding situations, reducing the risk of accidents or incidents.
- Better allocation of resources for crowd control and emergency response.

Success Criteria :

- Accurate detection and estimation of crowd density.
- Minimal latency in processing video feeds and triggering auditory alerts.
- High reliability and robustness across different environments and crowd densities.
- Positive feedback from stakeholders regarding the effectiveness of the system in managing overcrowding.

By addressing the outlined objectives and developing a comprehensive real-time crowd detection system, this project aims to contribute to safer and more efficient crowd management practices using advanced technology.

1.3.OBJECTIVE OF THE PROJECT:

The primary objective of the project "Real-Time Crowd Detection for Overcrowding Management Using OpenCV and pyttsx3" is to develop a comprehensive system capable of accurately detecting, monitoring, and managing crowd density in real-time. Specifically, the project aims to achieve the following objectives:

1. **Develop a Robust Crowd Detection Algorithm**: Create an algorithm leveraging OpenCV to accurately detect and estimate crowd density in real-time from video feeds. This algorithm should be capable of handling varying lighting conditions, crowd densities, and environmental factors.
2. **Integrate Auditory Alert System**: Utilize pyttsx3 to integrate real-time auditory alerts into the system. When overcrowding is detected, the system should promptly generate audible notifications to alert authorities or designated personnel, enabling timely intervention and crowd management.
3. **Ensure Real-Time Processing**: Implement the system to process video feeds in real-time with minimal latency. Timely processing is critical for enabling prompt responses to overcrowding situations and ensuring effective crowd management.
4. **Enhance Robustness and Adaptability**: Design the system to be robust

against potential challenges such as occlusions, varying camera angles, and dynamic crowd behaviors. Additionally, ensure adaptability to different types of environments and crowd scenarios to enhance the system's versatility and effectiveness.

5. **Validate Accuracy and Reliability**: Conduct comprehensive testing to validate the accuracy, efficiency, and reliability of the system. Testing should cover various real-world scenarios, including different environments, crowd densities, and lighting conditions, to ensure the system performs optimanlly in diverse settings

6. **Demonstrate Effectiveness**: Showcase the effectiveness of the system in enhancing crowd management and safety measures. This includes evaluating the system's performance in detecting overcrowding incidents, triggering timely alerts, and facilitating proactive crowd control measures to mitigate potential risks and ensure public safety. By achieving these objectives, the project aims to develop a robust and efficient real-time crowd detection system that contributes to improvedcrowd management practices, enhances safety measures in public spaces, and enables authorities to respond effectively to overcrowding incidents.

2.LITERATURE SURVEY

2.1 EXISTING SYSTEM:

Maniruzzaman et al. [19] presented a hypothetical Network of Everything (IoT)-based diabetics self-monitoring system which employs BLE devices to collect vital data, such blood glucose and weight. Using MongoDB for data storage and Apache Kafka for streaming messages and data, the system examines data in real time. The system's goals are to supply suggestions, treatment, and results to manage diabetes.Cappon et al. [19] evaluated the characteristics of currently in use commercial versions of CGM wearable sensors and also prototypes. A method for using mobile phones and smartwatches to track blood glucose, exercise, insulin injections, and nutritional data is described by Arsand et al. [19]. In an effort to efficiently handle the conditions of diabetic patients, Lee and Yoo 4 generated an arrangement that makes use of a PDA to track their blood pressure, blood glucose levels, food consumption, and routines of exercise. Rodriguez et al. [19] proposed a smartphone application that uses an automated glucometer to gather data from sensors. They also discussed the possibility of using big data analysis from sensors to address issues related to diabetes. Several academic institutions, like the School of Computer Sciences at University Sains Malaysia and the School of Computing and Engineering at the University of West London, are associated with Muhammad Ashir Ali et al. [19] Hafiz Husnain Raza Sherazi, Sukumar Letchmunan, and Umair Muneer Butt are the corresponding authors. In their research paper, J. I. Su´arez et al. [19] talked about using cellphones and a Bluetooth gas detecting module to monitor air quality.The study article 12 discusses gathering vital health information, such as weight, blood pressure, blood glucose, and heartbeat, from sensors such BLE wireless devices. Realtime processing of this data and demographic data is used to monitor health conditions and predict the risk of diabetes. A variety of approaches and strategies for tracking and identifying humans, such as particle filtering framework for multiperson tracking-by-detection, HOG-based head-shoulder recognition, and statistical features of facial skin colour. Additionally, they learnt how to apply CNNs and SVMs, two deep learning and machine learning techniques, for object detection. Furthermore, students learned how to train Haar cascade classifiers for human detection and evaluate the separation between people when using Raspberry Pi and OpenCV-Python for crowd control. Syed Ameer

Abbas, M. Anitha, and X. Vinitha Jaini's "Realisation of Multiple Human Head Detection and Direction Movement Using Raspberry Pi":

This study focuses on recognising human heads and counting the number of individuals in a particular region using OpenCV-Python and a Raspberry Pi 3 board. The authors of "People Counting System using Raspberry Pi" are Shim Jaechang and Md. Israfil Ansari. This study suggests utilising a Raspberry Pi and OpenCV-Python combination to create a low-cost method of tracking and counting people based on blob recognition. Jaysri Thangam, Padmini Thupalli Siva, and B. Yogameena's article "Crowd Video Count in Low Resolution Surveillance Head Detector and Colour based using Segmentation for Disaster Management" This study counts the number of individuals in a crowd, especially when there is occlusion, by using a colorbased segmentation technique and a general head detector.[13] Kanchan Mangrule, H. T., "Literature Survey of IoT Capable Crowd Analysis Using Raspberry Pi-3": Insights into IoTenabled crowd analysis with a Raspberry Pi-3 are presented in this study, emphasising the significance of controlling and monitoring big gatherings in real time.[13] Using technologies like YOLOv5 and DeepSort, the project focuses on creating software for item detection, person tracking, and crowd counting. From YOLOv2 to YOLOv5, the YOLO (You Only Look Once) algorithm has improved, with YOLOv5 reaching greater frames per second (FPS) than YOLOv4. The system makes use of PyTorch for deep learning tasks, Tkinter for UI creation, and OpenCV for image processing and camera access. Obstacles in Crowd Analysis: Obstacles in crowd analysis include low resolution, intra- and inter-scene variations, excessive clutter, contrast variations, uneven people distribution, and uneven lighting. Current Methods for Crowd Analysis: Li et al. examined various techniques for crowd scene analysis, including crowd motion pattern learning, crowd behaviour, activity analysis, and anomaly detection in crowds. Zhan et al. and Junior et al. investigated and reviewed current methods for general crowd analysis. Comparing Crowd Counting Techniques: Loy et al. provide a thorough explanation and analysis of crowd counting techniques based on video pictures, emphasising how successful CNN-based techniques are at lowering error rates. Methods for the Crowd Detection System: Three main categories of methods can be used to categorise the methods: detection-based, regression-based, and density-based. Detection-Based Approaches: Detection-based approaches use

simultaneous identification of individuals and their locations in order to estimate the population. For this, a variety of techniques and algorithms have been employed, including the Adaboost learning algorithm and the scanning window pedestrian detector. Regression-Based Methods: Regression-based methods extract feature mappings for crowd counting by working on local picture patches. For crowd counting, many characteristics are used, such as the Local Binary Pattern (LBP), the Histogram of Oriented Gradients (HOG), and regression approaches including neural networks and linear regression. Crowd Counting Systems: These systems use a variety of techniques, including regression-based estimation and detection, to identify and categorise people in crowd scenes before using classifiers to count them.

2.2 DISADVANTAGES OF EXISTING SYSTEM:

a) **Accuracy Limitations**:

- **Complex Environments**: OpenCV-based systems might struggle with accuracy in complex environments with variable lighting, shadows, and occlusions, which can hinder the reliable detection and counting of individuals in a crowd.
- **Diverse Crowd Characteristics**: Variability in crowd density,movement patterns, and individual appearances can pose challenges for consistent and accurate detection.

b) **Computational Resources**:

- **High Processing Power**: Real-time video processing requires significant computational resources, including powerful CPUs or GPUs, which can be costly and may not be feasible for deployment in all settings.
- **Scalability Issues**: Scaling up to monitor large areas or multiple locations simultaneously can lead to increased hardware and infrastructure costs.

c) **Latency and Real-Time Processing**:

Delay in Detection: There can be a delay between video capture, processing, and response, which may not be acceptable in scenarios where immediate action is required.

- **Network Dependence**: Systems that rely on network connections for data transmission can experience latency due to network issues, affecting real-time performance.

d) **Privacy Concerns**:

- **Surveillance**: Continuous video monitoring can raise privacy issues, as individuals may be uncomfortable with being constantly observed and recorded.
- **Data Security**: Storing and processing video data involves risks related to data breaches and unauthorized access, necessitating robust security measures.

e) **Integration and Maintenance**:

- **Complex Setup**: Setting up and configuring these systems can be technically challenging, requiring specialized knowledge in computer vision, software integration, and hardware setup.
- **Maintenance Overhead**: Regular maintenance is required to ensure the system's accuracy andreliability, including software updates, calibration, and troubleshooting.

f) **Environmental Constraints:**

- **Outdoor vs. Indoor**: Environmental factors such as weather conditions, lighting variations, and physical obstructions can significantly impact the performance of crowd detection systems, especially in outdoor settings.
- **Camera Positioning**: Optimal camera placement is crucial for effective monitoring, which may not always be possible due to architectural constraints or other practical limitations.

g) **False Positives and Negatives**:

- **Detection Errors**: The system might generate false positives (detecting a crowd when there isn't one) or false negatives (failing to detect an actual crowd), leading to unreliable crowd management decisions.

Dynamic Scenes: Rapidly changing scenes, such as sudden movements or large groups dispersing quickly, canchallenge the system's ability to maintain

In an era of rapid urbanization and increasing population density, managing crowds effectively has become a crucial aspect of public safety and infrastructure management. Whether it's ensuring social distancing in public spaces, monitoring crowd density at events, or optimizing traffic flow in transportation hubs, real-time crowd detection systems play a vital role.

This project aims to develop a real-time crowd detection system utilizing OpenCV (Open Source Computer Vision Library) and pyttsx3 (a Python library for text-to-speech conversion) to address the challenges of overcrowding management. By leveraging computer vision techniques, the system can analyze live video feeds from cameras installed in various locations and provide insights into crowd density levels.

The integration of OpenCV allows for robust image processing and object detection capabilities, enabling the system to identify and track individuals within a crowd. Through the utilization of machine learning algorithms such as Haar cascades or more advanced deep learning models like YOLO (You Only Look Once), the system can accurately detect and count the number of people present in a given area.

Additionally, pyttsx3 facilitates real-time audio feedback, enabling the system to communicate crowd density information audibly. This feature is particularly beneficial in scenarios where visual monitoring may not be feasible or when immediate action is required to manage overcrowding situations.

The proposed system offers several potential applications across various domains, including:

1. **Public Safety:** Enhancing crowd management protocols in public spaces such as stadiums, airports, train stations, and shopping malls to ensure compliance with occupancy limits and social distancing guidelines.
2. **Event Management**: Providing event organizers with real-time insights into crowd density levels to optimize venue layout, manage queues, and ensure the safety and comfort of attendees.
3. **Traffic Management**: Monitoring pedestrian flow in urban areas to improve traffic management strategies, reduce congestion, and enhance pedestrian safety.
4. **Emergency Response**: Facilitating rapid response to emergencies by providing authorities with timely information on crowd movements and density, enabling effective evacuation procedures and resource allocation.

Overall, by combining the power of computer vision with audio feedback capabilities, the proposed crowd detection system offers a versatile and effective solution for overcrowding management in various real-world scenarios

2.3 PROPOSED SYSTEM:

The proposed system is designed to monitor and manage crowd density in real-time by utilizing computer vision and audio feedback technologies. By integrating OpenCV, Haar Cascade classifiers, and Pyttsx3, the system provides an automated solution for detecting overcrowding and issuing timely alerts to ensure safety and compliance with regulations. Below is a detailed breakdown of the proposed system:

1. System Overview

The system consists of three primary components:

Crowd Detection Module: Utilizes OpenCV and Haar Cascade classifiers to analyze video feeds and detect individuals in the monitored area.

Alert Generation Module: Employs Pyttsx3 to generate audible alerts when crowd density exceeds predefined thresholds.

Processing Framework: A software pipeline that integrates the detection and alert

modules, processes video streams, and triggers alerts in real-time.

2. System Workflow

The proposed system operates through the following steps:

Video Capture:

A camera is used to capture live video feeds of the area to be monitored. This can be a CCTV camera, webcam, or any compatible video capturing device.

The video feed is continuously streamed into the system for processing.

Preprocessing:

The video frames are extracted from the stream and converted to grayscale to reduce computational complexity.

Noise reduction techniques, such as Gaussian blur, are applied to enhance the quality of the input data and improve detection accuracy.

Crowd Detection:

Haar Cascade Classifiers are used to identify human features, such as upper bodies or faces, in each video frame.

The system scans the frame using a sliding window technique to detect patterns matching the trained Haar Cascade model.

Detected objects (people) are highlighted with bounding boxes, and the total count is updated.

Threshold Analysis:

The system compares the detected crowd count with a predefined threshold value.

If the count exceeds the threshold, the system identifies the situation as overcrowding.

Alert Triggering:

When overcrowding is detected, the Pyttsx3 library is used to generate a voice alert.

The audio alert communicates a warning, such as “Attention: The area is overcrowded. Please disperse to maintain safety.”

The alert can be customized based on the scenario or environment.

Continuous Monitoring:

The system runs in a loop to ensure real-time detection and alerting, continuously analyzing video streams and responding to changes in crowd density.

3. Key Components

OpenCV (Computer Vision Library):

OpenCV is the primary tool for processing video feeds and detecting individuals.

Its pre-trained Haar Cascade classifiers enable rapid and efficient detection of human features in images or video frames.

Haar Cascade Classifiers:

These are lightweight and pre-trained models designed to detect objects (e.g., faces, upper bodies) using pattern recognition.

They are ideal for real-time applications due to their low computational requirements.

Pyttsx3 (Text-to-Speech Library):

Pyttsx3 is used to convert text into speech for generating voice alerts.

It allows customization of the voice output, including volume, speed, and language, ensuring flexibility for various applications.

Hardware:

A camera device for capturing video feeds.

A speaker or audio system for broadcasting alerts.

4. Features of the Proposed System

Real-Time Detection:

The system processes video streams in real-time, ensuring immediate detection of overcrowding.

Automated Alerts:

Alerts are triggered automatically without the need for manual intervention, reducing response time.

Customizable Thresholds:

The crowd limit threshold can be configured based on the size and capacity of the monitored area.

Scalability:

The system can be scaled up to monitor multiple areas simultaneously by integrating additional cameras and processing units.

Cost-Effective:

The use of open-source libraries (OpenCV, Pyttsx3) and readily available hardware makes the system affordable and accessible.

5. Challenges and Limitations

Environmental Factors:

Variations in lighting, weather conditions, or background noise can impact the accuracy of detection.

For example, poor lighting may reduce the visibility of individuals, leading to false negatives.

Detection Accuracy:

Haar Cascade classifiers, while efficient, may struggle with overlapping objects or crowded environments, leading to false positives or missed detections.

Hardware Dependence:

The quality of the camera and processing unit directly affects the system's performance. Low-resolution cameras may limit detection accuracy.

Scalability Issues:

Monitoring large areas or multiple zones simultaneously may require significant computational resources.

6. Potential Improvements

To enhance the performance and robustness of the proposed system, the following improvements can be considered:

Integration of Deep Learning Models:

Advanced object detection models such as YOLO (You Only Look Once) or SSD (Single Shot Multibox Detector) can improve detection accuracy in complex environments.

These models are better suited for detecting multiple individuals in crowded or cluttered scenes.

Multi-Modal Sensors:

Combining visual data with other sensors, such as thermal cameras or infrared sensors, can enhance detection capabilities, especially in low-light conditions.

Predictive Analytics:

Incorporating machine learning algorithms to predict crowd density trends based on historical data and real-time observations.

Mobile Integration:

Developing a mobile application or web-based interface for remote monitoring and alert management.

Smart Notifications:

Alerts can be sent via SMS, email, or mobile notifications to authorities for faster response.

7. Applications of the System

Public Spaces:

Ensuring safety in parks, malls, transportation hubs, and other high-traffic areas.

Event Management:

Monitoring crowd density during concerts, festivals, or sports events to prevent overcrowding.

Critical Infrastructure:

Enhancing security and crowd control in airports, railway stations, and other sensitive locations.

Emergency Response:

Facilitating evacuation planning and resource allocation during emergencies or disasters.

8. Benefits of the Proposed System

Improved Public Safety:

Reduces risks associated with overcrowding, such as stampedes or accidents.

Operational Efficiency:

Automates the process of crowd monitoring, freeing up human resources for

other tasks.

Timely Interventions:

Real-time alerts enable faster decision-making and preventive actions.

Cost-Effective Solution:

Utilizes open-source technologies and existing hardware, making it affordable for wide-scale deployment.

2.4 ADVANTAGES OF PROPOSED SYSTEM

Alerting System

The proposed system offers numerous benefits, making it an innovative and practical solution for real-time crowd monitoring and management. Below are the key advantages:

1. Enhanced Public Safety

The system ensures timely detection of overcrowding, reducing risks such as stampedes, accidents, or suffocation in densely populated areas.

By providing real-time alerts, it enables authorities to take preventive measures to maintain a safe environment.

2. Automation and Efficiency

The system eliminates the need for manual crowd monitoring, reducing human effort and errors.

It provides continuous monitoring, ensuring no lapse in detection and response.

3. Real-Time Alerts

The integration of audio alerts using Pyttsx3 ensures immediate communication with the crowd, facilitating quick action to manage crowd density.

Alerts are automated, ensuring that warnings are consistent and unbiased.

4. Cost-Effectiveness

The use of open-source libraries like OpenCV and Pyttsx3 makes the system affordable to develop and implement.

It can utilize existing cameras and hardware, reducing additional infrastructure costs.

5. Scalability

The system can be easily scaled to monitor multiple areas by adding more cameras or processing units.

It is adaptable for various environments, from small indoor spaces to large public areas.

6. Versatility

The system has diverse applications, including event management, public safety in transportation hubs, shopping malls, parks, and emergency response scenarios.

Its customizable features allow it to meet the specific needs of different use cases.

7. Fast and Accurate Detection

The use of Haar Cascade classifiers allows for rapid identification of individuals in video feeds, enabling the system to react in real-time.

Advanced algorithms ensure reliable detection under normal conditions.

8. Improved Crowd Behavior Management

Audible alerts guide the crowd to disperse or move to less crowded areas, promoting organized behavior.

It helps reduce panic in crowded scenarios by providing clear and authoritative instructions.

9. Inclusivity

Audio alerts ensure that individuals with visual impairments or those not paying attention to visual displays can still receive critical warnings.

10. Customization

Threshold values for crowd density can be adjusted based on the size and capacity of the monitored area.

Alerts can be customized for different languages, volume levels, and specific instructions.

11. Low Maintenance

The system's software is easy to maintain, and updates or enhancements can be

implemented without significant downtime.

The use of durable hardware like cameras and speakers ensures long-term usability.

12. Preventive Decision-Making

By providing real-time insights into crowd density, the system helps authorities make proactive decisions to prevent overcrowding and associated risks.

It supports better planning and resource allocation for large events or emergencies.

13. Environment-Friendly

The system reduces the need for printed signs or manual staff interventions, aligning with sustainable and eco-friendly practices.

3.SYSTEM ANALYSIS

3.1 PURPOSE:

The persistent issue of overcrowding in public spaces, especially during events, festivals, and gatherings, presents a significant challenge for effective crowd management. Ensuring the safety and comfort of individuals while adhering to crowd regulations necessitates the implementation of innovative and efficient solutions. This project aims to address these challenges by developing a real-time crowd limitation alerting system that leverages cutting-edge technologies, including OpenCV, Haar Cascade classifiers, and Pyttsx3. The system's primary objective is to provide an automated, accurate, and cost-effective method for monitoring crowd density and issuing timely alerts to prevent overcrowding and associated risks.

The core purpose of this project is to create a solution that enhances public safety and reduces the need for manual interventions in monitoring large crowds. Overcrowding can lead to various risks, including accidents, stampedes, and discomfort, which can escalate into critical situations if not addressed promptly. By employing a real-time monitoring system, authorities and event organizers can detect crowd surges early and take preventive measures to ensure a safe environment for attendees. Furthermore, the system integrates audio feedback through Pyttsx3, making it possible to deliver clear and audible alerts to inform people about overcrowding or guide them to safer areas.

Importance of Automation in Crowd Management

Manual crowd management often requires significant manpower and can be prone to delays and human error, particularly in large-scale events. Automating this process ensures faster detection and response to overcrowding, minimizing risks associated with delayed interventions. The integration of OpenCV with Haar Cascade classifiers allows the system to process video feeds and accurately detect the number of people in a designated area. This real-time detection is crucial for environments where crowd density can change rapidly, such as stadiums, religious gatherings, or music festivals.

The system is designed to continuously monitor the area using a camera feed, analyze the video

frames, and determine whether the crowd exceeds a predefined threshold. When this threshold is breached, the system triggers an audio alert generated using Pyttsx3. This automated process eliminates the reliance on manual monitoring and ensures that alerts are generated consistently and without bias, enabling authorities to act promptly and effectively.

Leveraging OpenCV and Haar Cascade Classifiers

OpenCV, an open-source computer vision library, serves as the backbone of this project's real-time crowd detection capabilities. Its flexibility and extensive library of tools make it suitable for processing video streams and implementing object detection algorithms. Haar Cascade classifiers, which are pre-trained models within OpenCV, are employed to detect human features such as upper bodies or faces in video frames. These classifiers work by scanning the video feed for patterns that match the features of interest, allowing for rapid and efficient detection of people.

The use of Haar Cascade classifiers is particularly advantageous for scenarios where computational resources are limited. These classifiers are lightweight and can process video frames in real-time, making them ideal for environments where high-speed detection is required. Additionally, the modularity of OpenCV allows for future enhancements, such as the integration of more advanced detection algorithms like deep learning-based models, to improve accuracy in challenging conditions.

Enhancing Communication with Pyttsx3

In addition to detecting overcrowding, the system incorporates Pyttsx3, a text-to-speech (TTS) library, to deliver audible alerts. Audio feedback is a critical component of crowd management, as it ensures that alerts are accessible to a wide audience, including those who may not be actively monitoring visual displays or notifications. The use of Pyttsx3 enables the system to generate clear and customizable voice messages, which can be tailored to specific scenarios or locations.

For instance, in the event of overcrowding, the system can issue warnings such as "Attention: The area is overcrowded. Please disperse to maintain safety." This immediate feedback helps to manage crowd behavior proactively, reducing the likelihood of panic or disorder. The integration

of Pyttsx3 also ensures inclusivity, as audible alerts can reach individuals who may have visual impairments or are not actively observing visual cues.

Addressing Challenges in Crowd Detection

While the proposed system offers numerous advantages, it also faces certain challenges that must be addressed to ensure optimal performance. One of the primary challenges is the accuracy of Haar Cascade classifiers in complex environments. Factors such as poor lighting, occlusions, or overlapping individuals can reduce the effectiveness of the detection algorithm, leading to false positives or negatives. These inaccuracies may result in missed alerts or unnecessary warnings, which could undermine the system's reliability.

To mitigate these challenges, the system can be enhanced with additional technologies. For example, the integration of deep learning-based object detection models such as YOLO (You Only Look Once) or MobileNet-SSD can significantly improve accuracy by providing more robust and adaptable detection capabilities. These models are capable of handling diverse environmental conditions and can differentiate between humans and other objects with greater precision. Additionally, combining visual data with other sensors, such as thermal cameras or motion detectors, can further enhance the system's ability to detect and track individuals accurately.

Applications of the System

The real-time crowd limitation alerting system has a wide range of applications across various domains. In public spaces such as shopping malls, transportation hubs, or parks, the system can be used to monitor crowd density and ensure compliance with safety regulations. During large-scale events, such as concerts, sports matches, or religious gatherings, the system can help organizers manage crowd flow and prevent dangerous overcrowding.

Moreover, the system can be deployed in critical infrastructure, such as airports or railway stations, to enhance security and streamline crowd movement. By providing real-time insights

into crowd density, authorities can make informed decisions about resource allocation, emergency response, or evacuation procedures. The system's flexibility and scalability make it suitable for both small-scale and large-scale implementations, ensuring its relevance in diverse settings.

Contributions to Public Safety and Efficiency

This project's primary contribution lies in its ability to enhance public safety by providing a reliable and automated method for monitoring and managing crowd density. The integration of computer vision and audio feedback technologies ensures that alerts are generated in real-time, enabling timely interventions that can prevent accidents or disruptions. Additionally, the system reduces the reliance on manual labor, allowing resources to be allocated more efficiently.

By addressing the challenges of overcrowding, the system promotes a safer and more organized environment for individuals in public spaces. Its modular design and reliance on open-source libraries also make it a cost-effective solution that can be customized to meet the specific needs of different locations or scenarios. The project demonstrates the potential of technology to address real-world challenges and underscores the importance of innovation in improving public safety and operational efficiency.

Future Scope and Enhancements

The current implementation of the system serves as a foundational framework for real-time crowd monitoring and alerting. However, there are several opportunities for future enhancements. The integration of advanced deep learning models can improve detection accuracy and enable the system to handle more complex scenarios. Additionally, incorporating multi-modal sensors, such as audio or thermal sensors, can provide additional layers of data for more comprehensive analysis.

The system can also be extended to include features such as predictive analytics, which can forecast crowd density trends based on historical data and current observations. This capability would allow authorities to take proactive measures to manage crowd flow and prevent overcrowding. Furthermore, the development of a user-friendly interface or mobile application can enhance the accessibility and usability of the system, making it easier for operators to

monitor and respond to alerts.

3.2 REQUIREMENT ANALYSIS

3.2.1 Functional Requirements

1. **Real-Time Video Processing**:

   - Capture and process video feeds from multiple cameras in real-time.
   - Support various video resolutions and frame rates.

2. **Crowd Detection**:

   - Accurately detect individuals in the video feed using computer vision techniques.
   - Handle different crowd densities and dynamically changing environments.

3. **Crowd Counting**:

   - Count the number of individuals in designated areas.
   - Maintain accuracy under varying conditions such as lighting changes, shadows, and partial occlusions.

4. **Alerts and Notifications**:

   - Generate alerts when crowd density exceeds predefined thresholds.
   - Customize alerts based on different thresholds for various scenarios.

5. **Text-to-Speech (TTS) Output**:

   - Use pyttsx3 to provide real-time audible alerts and instructions.
   - Ensure TTS is clear and understandable, even in noisy environments.

6. **Data Logging and Reporting:**

Log crowd data for analysis and reporting purposes

Provide access to historical data for post-event analysis and decision-making.

3.2.2Non- Functional Requirements:

1. **Performance**:

   - Process video feeds with minimal latency for timely detection and response.
   - Support real-time operation without significant delays.

2. **Scalability**:

   - Monitor multiple locations and large areas simultaneously.
   - Handle increased video feeds without performance degradation.

3. **Reliability**:

   - Ensure continuous operation without failures.
   - Include fault detection and recovery mechanisms.

4. **Usability**:

   - Provide an intuitive user interface for operators with varying technical expertise.
   - Allow easy configuration and adjustment of detection parameters.

5. **Compatibility**:

   - Be compatible with various camera models and video input sources.
   - Work across different operating systems supported by OpenCV and pyttsx3.

6. **Security**:

   - Ensure the security and privacy of video feeds and stored data.
   - Implement access control mechanisms to restrict unauthorized access.

7. **Environmental Adaptability**:

- Perform reliably in diverse indoor and outdoor environments.
- Adapt to changes in lighting, weather conditions ,environmental Factors 3without performance degradation

8. **Reliability**:

- Ensure continuous operation without failures.
- Include fault detection and recovery mechanisms.

9. **Usability**:

- Provide an intuitive user interface for operators with varying technical expertise.
- Allow easy configuration and adjustment of detection parameters.

10. **Compatibility**:

- Be compatible with various camera models and video input sources.
- Work across different operating systems supported by OpenCV and pyttsx3.

11. **Security**:

- Ensure the security and privacy of video feeds and stored data.
- Implement access control mechanisms to restrict unauthorized access.

12. **Environmental Adaptability**:

- Perform reliably in diverse indoor and outdoor environments.

Adapt to changes in lighting, weather conditions, and other environmental factors.

3.3 REQUIREMENT SPECIFICATIONS

3.3.1 Hardware Requirements:

1. **Cameras**:

- High-definition cameras capable of capturing clear video feeds.
- Support for multiple camera models and configurations (fixed, PTZ, etc.).

Processing Unit:

- A computer with a multi-core processor (e.g., Intel i7 or AMD Ryzen 7) for efficient processing.
- A GPU (e.g., NVIDIA GTX 1080 or higher) to accelerate video processing and deep learning algorithms.

2. **Memory**:

- At least 16GB of RAM to handle real-time processing and multiple video streams.

3. **Storage**:

- SSD with sufficient capacity (e.g., 512GB or higher) for storing video data and logs.
- Additional external storage for archival purposes if necessary.

4. **Network**:

- High-speed network connectivity for streaming video feeds and remote monitoring.
- Secure communication channels to protect data transmission.

3.3.2 Software Requirements:

Operating System

3.3.2.1 Windows 10 or later, **macOS**, or **Linux** (Ubuntu recommended)

Programming Language

3.3.2.2 Python 3.7+

Python Libraries and Packages

1. **OpenCV**

   - For image and video processing.
   - Installation: pip install opencv-python

2. **NumPy**

   - For numerical operations.
   - Installation: pip install numpy

3. **pyttsx3**

   - For text-to-speech conversion.
   - Installation: pip install pyttsx3

4. **Imutils**

   - For simplifying OpenCV functions.
   - Installation: pip install imutils

5. **Scipy**

   - For additional scientific computations.
   - Installation: pip install scipy

6. **Matplotlib** (optional, for debugging and visualization)

- For plotting and visualization.
- Installation: pip install matplotlib

7. **dlib** (optional, if using facial landmarks for crowd detection)

- For face detection and shape prediction.
- Installation: pip install dlib

8. **TensorFlow/Keras or PyTorch** (optional)

- If you need a pre-trained deep learning model for more accurate crowd detection.
- TensorFlow Installation: pip install tensorflow
- PyTorch Installation: pip install torch torchvision

Development Tools

- **Integrated Development Environment (IDE)**
    - Recommended: PyCharm, VSCode, Jupyter Notebook

Additional Tools

- **FFmpeg**
    - For video processing and handling various video formats.

Installation: Follow the instructions on FFmpeg official website

4.SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE:

Modules

Initialization: **For auditory notifications, the script initialises a text-to-speech engine (using pyttsx3). Using OpenCV, Haar cascades for full-body and pedestrian detection are loaded.**

Video Capture: Using a given source (in this example, a video file called ”pep.mp4”), the script records video frames.

Detection and Tracking: To facilitate effective processing, the collected frames are transformed to grayscale. Haar cascades with parameters for scale factor, minimum neighbours, and minimum size are used to detect pedestrians and entire bodies. The bounding box area of each detected individual is used to group them, with larger groupings being given preference.

Grouping and Alerting: Using a predetermined threshold (group-threshold), the script keeps an eye out for the emergence of crowds. The text-to-speech engine generates and plays a message when it detects a crowd. For recognised groups, a visual sign is also shown on the video frames.

Queue Management: An area of interest that is designated as a ”queue-area” by the script is where individuals are tracked independently. On the video frames, the queue size is computed and shown as a status.

User Interface: The video stream shows visual information such as the number of people, the length of the queue, and alarm messages. In an OpenCV window, the video feed with the overlayed data is continuously displayed.

User Interaction:Hitting the ’q’ key will end the script.

- SSD with sufficient capacity (e.g., 512GB or higher) for storing video data and logs.
- Additional external storage for archival purposes if necessary.

5. **Network**:

- High-speed network connectivity for streaming video feeds and remote monitoring.
- Secure communication channels to protect data transmission.

ARCHITECTURE:

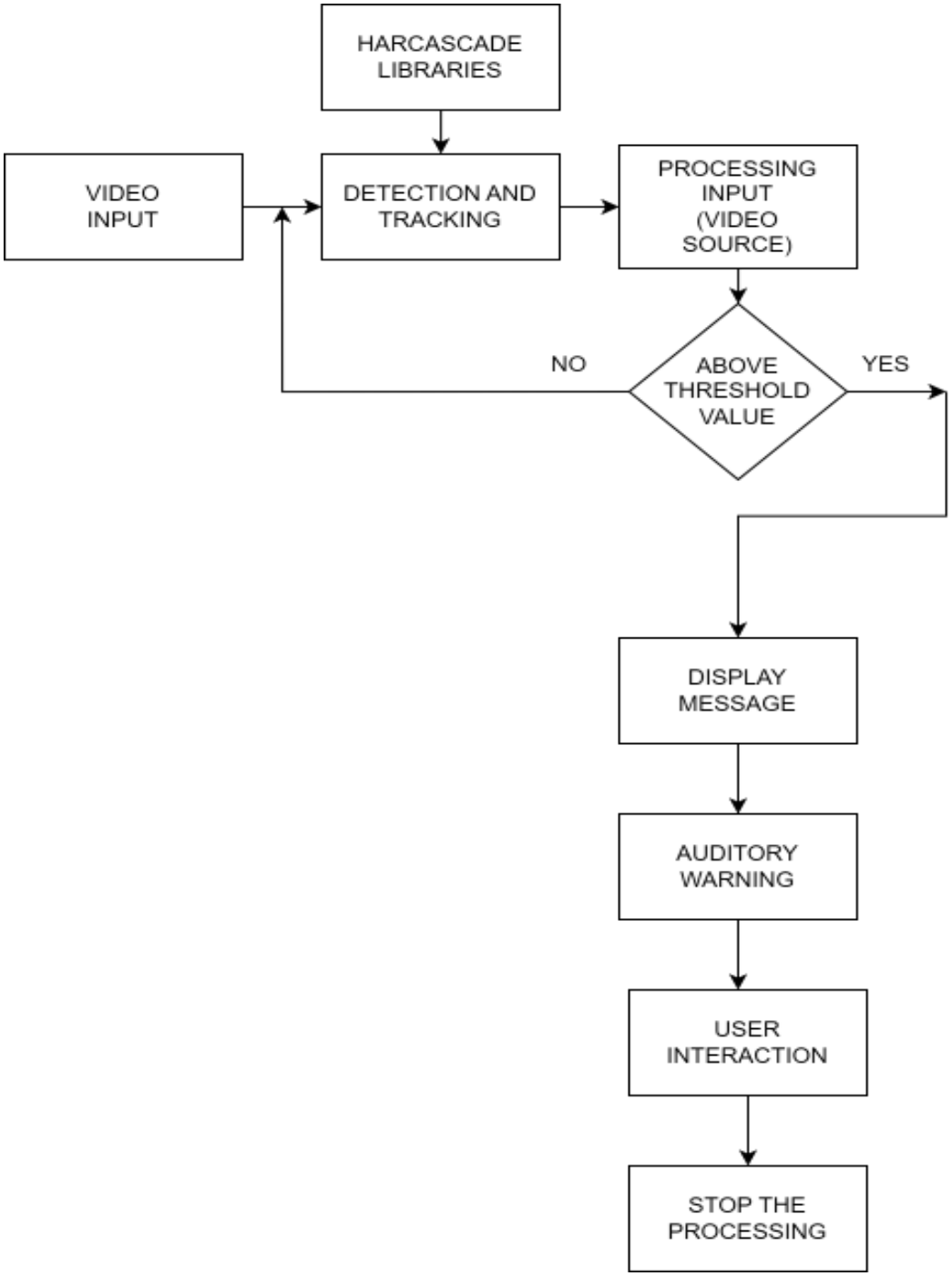


Fig:4.1.1

4.2 DESCRIPTION:

Opencv

OpenCV, brief for Open Source Computer Vision Library, is an open-source computer vision and machine learning program library. It was initially created by Intel in 1999 and has since ended up one of the foremost broadly utilized libraries for real-time computer vision applications. OpenCV is composed in C++ and offers interfacing for C++, Python, Java, and MATLAB, making it available to a wide gathering of people of developers.

The library gives a comprehensive set of apparatuses and calculations for different assignments in computer vision, picture handling, and machine learning. A few of the key highlights of OpenCV include:

Picture Preparing: OpenCV offers a wealthy set of capacities for fundamental and progressed picture preparing errands, such as sifting, changes, color space transformations, and geometric operations.

Question Discovery and Acknowledgment: It incorporates pre-trained models and calculations for question discovery, acknowledgment, and following. Procedures like Haar cascades, Hoard (Histogram of Arranged Angles), and profound learning-based strategies are supported.

Highlight Discovery and Portrayal: OpenCV gives capacities for recognizing and portraying key highlights in pictures, such as corners, edges, and keypoints. These highlights are significant for assignments like picture coordinating, stereo vision, and 3D reconstruction.

Machine Learning Integration: OpenCV consistently coordinating with prevalent machine learning systems like TensorFlow, PyTorch, and scikit-learn. It permits clients to construct and prepare custom models for different computer vision errands, counting classification, relapse, and clustering.

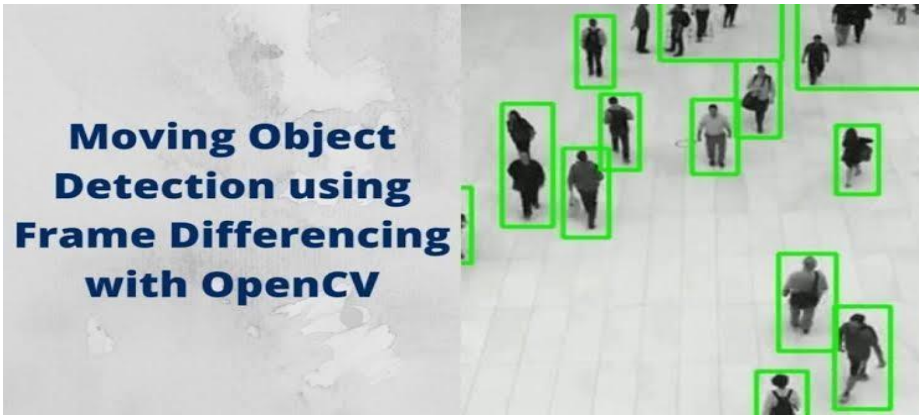


Fig. 4.2.1 Example Image of OpenCV

Table 1. OPEN CV

Component	Requirement
OpenCV Version	Compatible with system setup and dependencies.
Compiler	Visual Studio(Windows),GCC(Linux),etc.
GPU	NVDIA GPU with CUDA support(for GPU acceleration).
Dependencies	GTK,ffmpeg.
Python Packages	OpenCV,Numpy,Matplotlib,etc.

Camera Calibration and 3D Remaking: The library incorporates apparatuses for camera calibration, stereo vision, and 3D recreation, empowering applications like expanded reality, structure from movement, and profundity sensing.

Cross-Platform Compatibility: OpenCV is cross-platform and runs on different working frameworks counting Windows, Linux, macOS, Android, and iOS. This permits designers to send their applications over distinctive stages with ease.

Pyttsp

A bundle called PyTTSPy might be intended for text-tospeech (TTS) integration. It

provides an easy-to-use interface for converting written text into spoken language. PyTTS includes a variety of features, including the ability to choose from a variety of voices, adjust the speech rate, and create sound recordings from input content.

It makes use of various TTS motors and APIs, promot2ing flexibility in selecting the preferred blend technique. PyTTSPy is a useful tool for engineers that allows them to easily integrate text-to-speech functionality into their projects.

Haarcascade Classifier

Haar Cascade classifiers are a machine learning-based approach utilized for question discovery in pictures or recordings. They are especially prevalent for recognizing objects like faces, eyes, and other predefined designs. The strategy was presented by Paul Viola and Michael Jones in their seminal paper in 2001.

Here's a brief depiction of how Haar Cascade classifiers work:

Include Choice: Haar-like highlights are rectangular channels that are connected to an picture to extricate important data. These highlights are comparable to convolutional bits but are less complex and computationally efficient. Preparing: At first, a Haar Cascade classifier is prepared employing a expansive number of positive and negative samples. Positive tests are pictures containing the target question (e.g., faces), whereas negative tests are pictures that don't contain the protest. Amid preparing, the classifier learns to recognize between positive and negative tests by recognizing designs within the Haar-like features.

Cascade of Classifiers: The prepared classifier comprises of different stages organized in a cascade design. Each arrange comprises of a few powerless classifiers, which are straightforward choice capacities that classify a locale of the picture as either containing the question or not. The cascade structure permits for early dismissal of locales that are improbable to contain the protest, in this manner progressing efficiency.

Necessarily Picture: To speed up the computation of Haarlike highlights, an integral picture representation of the input picture is utilized. This representation permits for quick calculation of rectangular highlights at diverse scales and positions.

Sliding Window: The classifier is connected to the input picture employing a

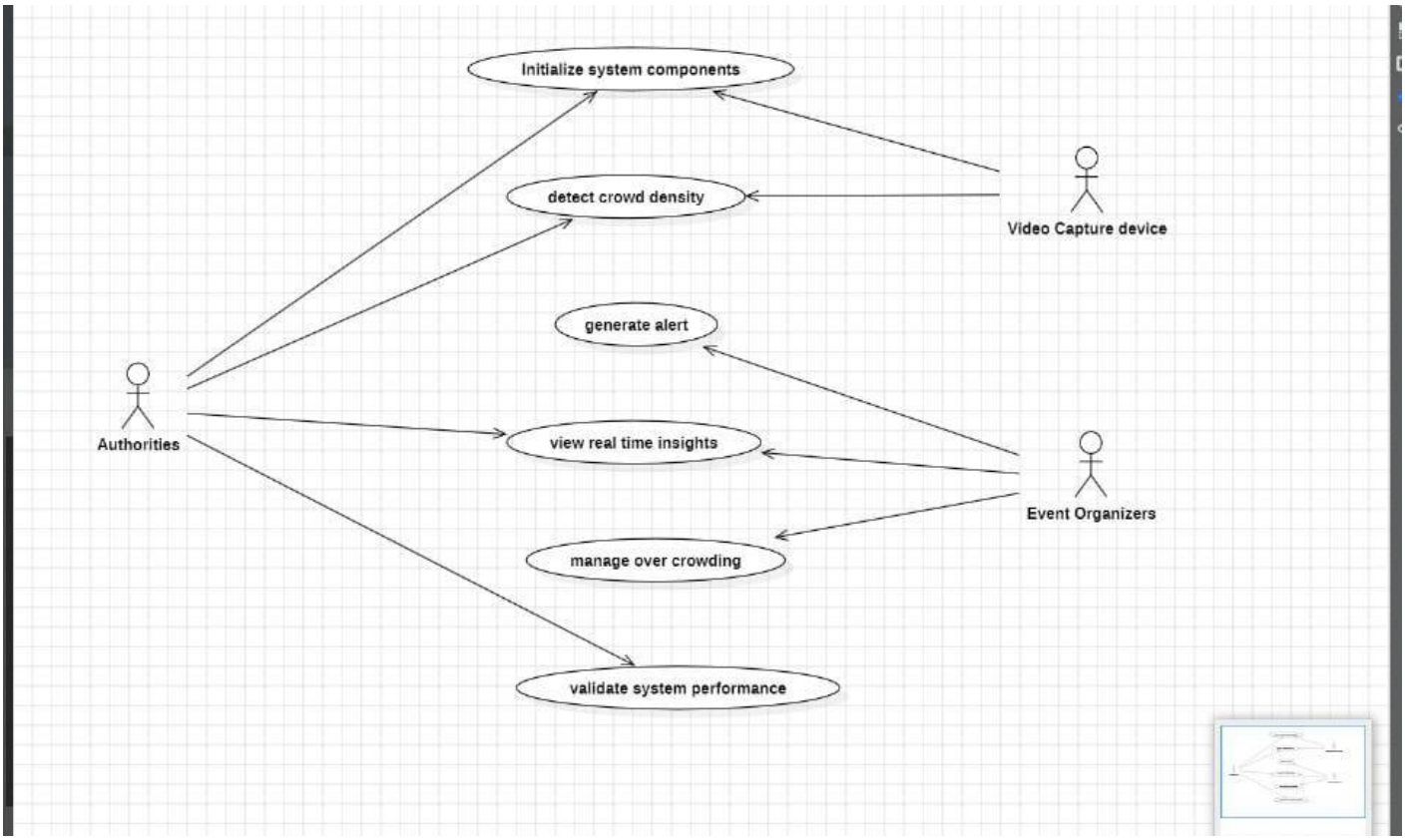
sliding window approach. The window moves over the picture, and at each position, the classifier assesses whether the locale inside the window contains the protest of interest.

Thresholding and Classification: The yield of each weak classifier is combined to form a final choice around whether the question is show within the locale being inspected. This choice is based on a edge, which is decided amid training. Haar Cascade classifiers are broadly utilized in different applications, counting confront discovery in photos, facial acknowledgment frameworks, pedestrian detection in independent vehicles, and more.

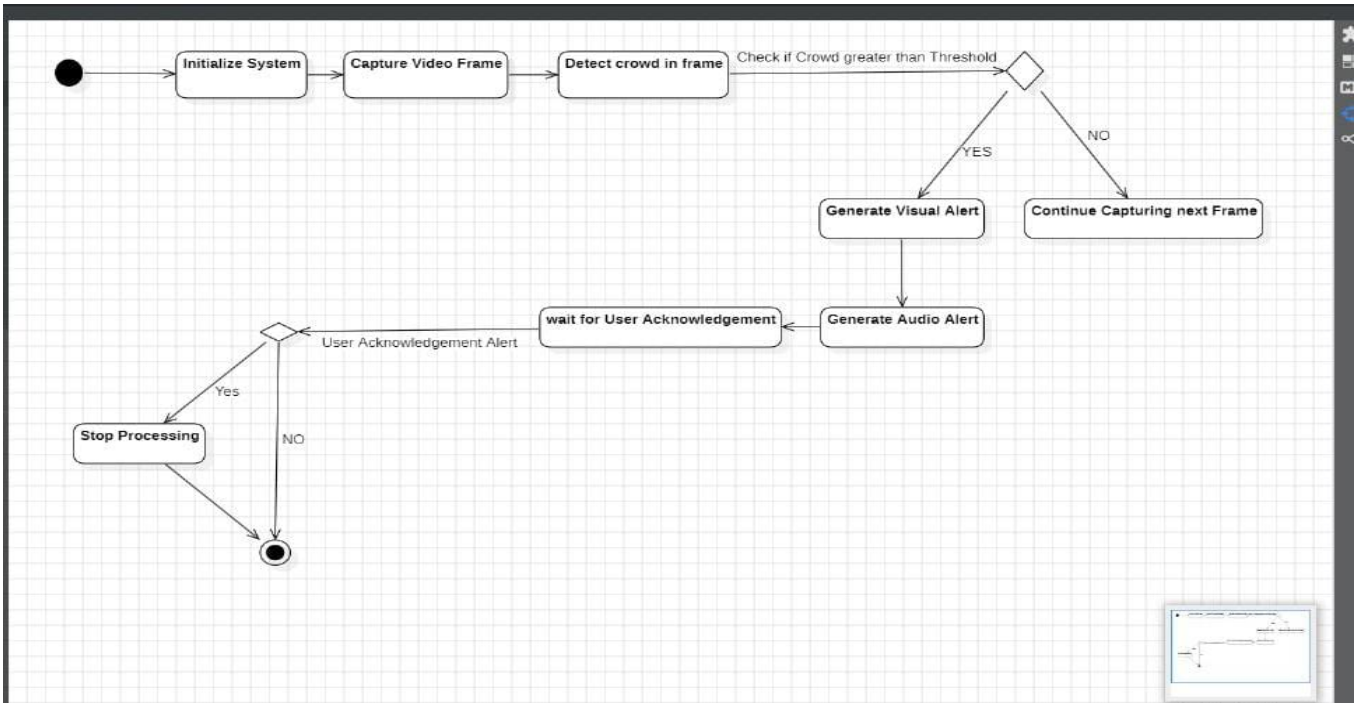
They offer a adjust between precision and computational productivity, making them appropriate for real-time applications. In any case, they may not perform well beneath challenging conditions such as occlusions, varieties in lighting, or changes inscale.

4.3 UML DIAGRAMS:

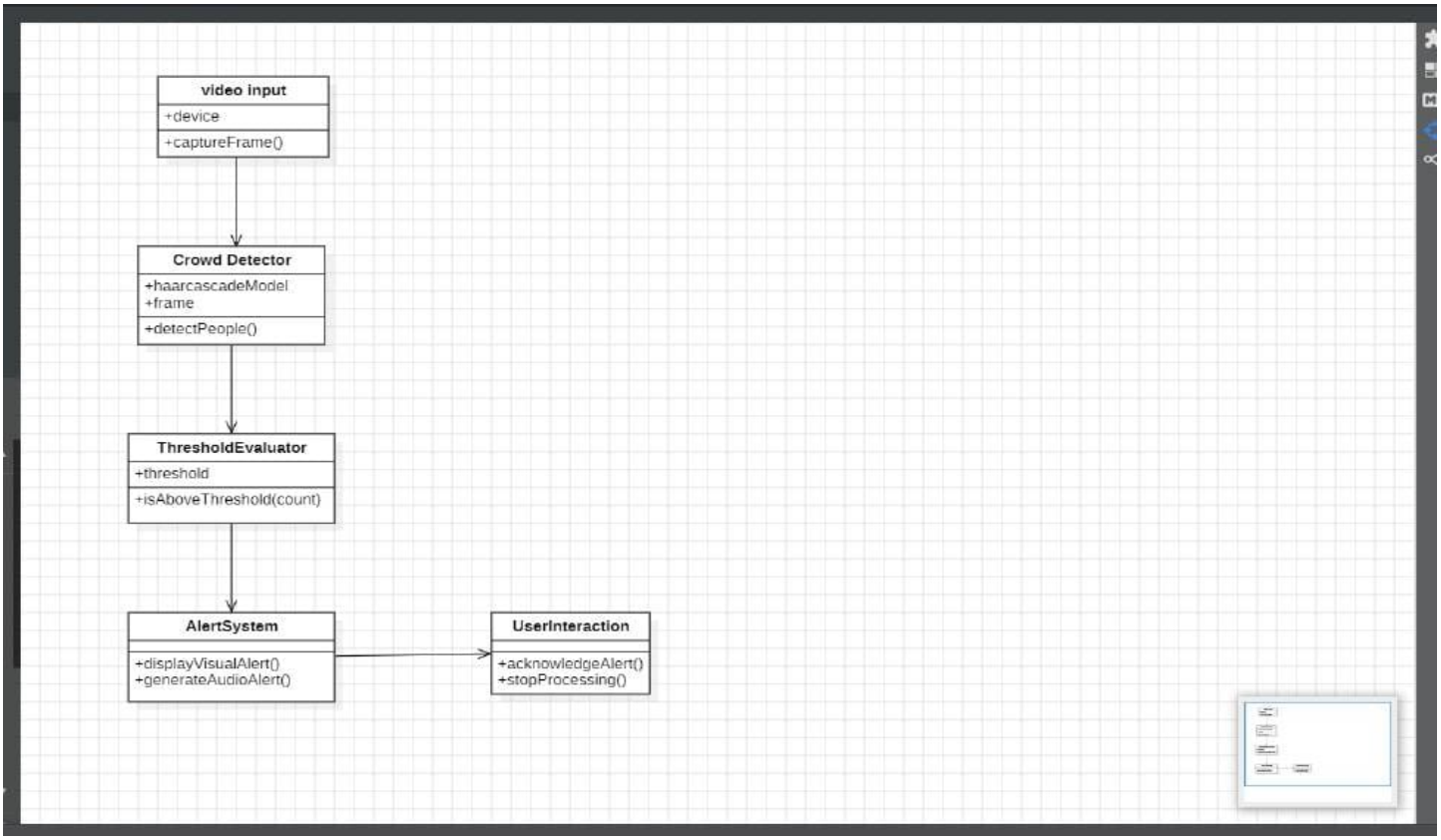
4.3.1.USECASE DIAGRAM:



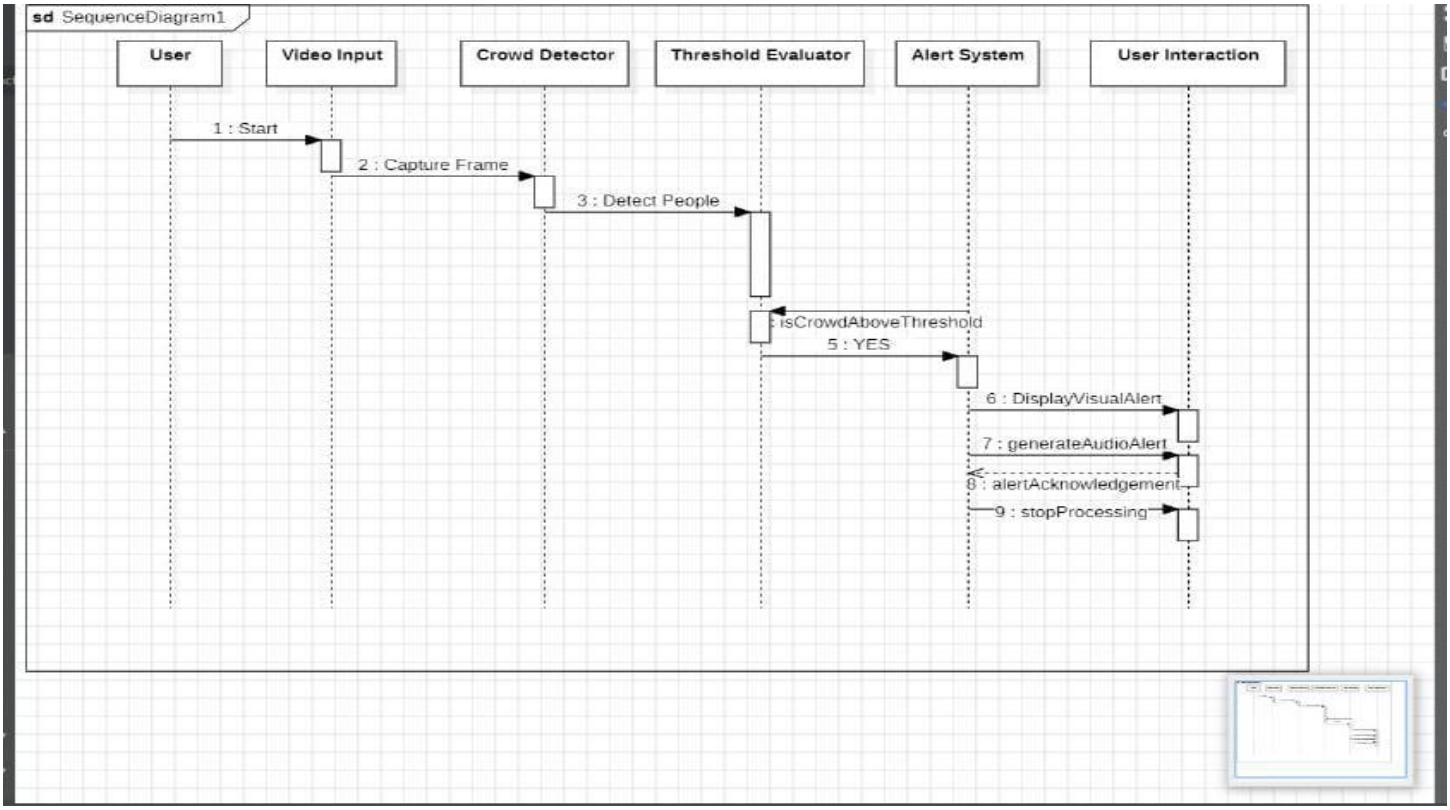
4.3.2 ACTIVITY DIAGRAM:



4.3.3 CLASS DIAGRAM:



4.3.4 SEQUENCE DIAGRAM



5.IMPLEMENTATION AND RESULTS

5.1 Method of Implementation:

The system leverages the OpenCV library for real-time image processing and crowd detection. The detection process involves the following steps:

1. **Video Capture**: Acquiring video streams from cameras positioned in the target area.
2. **Frame Processing**: Using OpenCV to process each video frame to detect and count individuals in the scene.
3. **Density Analysis**: Analyzing the number of detected individuals to determine crowd density.
4. **Alert Mechanism**: Utilizing pyttsx3 to generate and deliver audible alerts when the crowd density surpasses a critical threshold.

The experimental setup involves deploying the system in various environments with differing crowd densities to assess its performance. Key metrics include detection accuracy, response time, and the effectiveness of the alert mechanism.

Findings

The experimental analysis yielded the following findings:

1. **Detection Accuracy**: The system demonstrated high accuracy in detecting individuals in moderately crowded environments. However, accuracy decreases in highly dense crowds due to occlusions and overlapping individuals.
2. **Response Time**: The real-time processing capabilities of OpenCV allowed for prompt detection and alert generation, with minimal latency observed in video processing and speech synthesis.

3. **Alert Effectiveness**: The use of pyttsx3 for text-to-speech alerts proved effective in delivering clear and immediate warnings. This helped in prompt crowd management actions, potentially mitigating safety risks.
4. **Scalability**: The system's performance was consistent across different environments, indicating good scalability. However, hardware limitations such as camera resolution and processing power affected the overall efficiency.

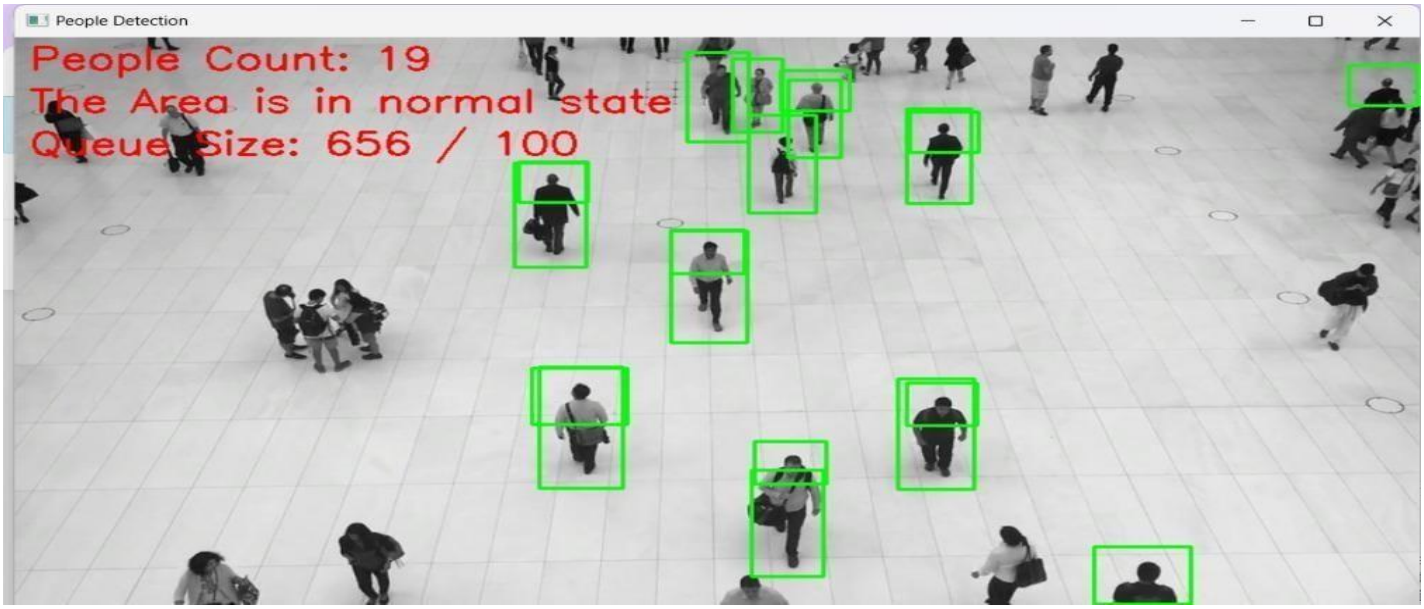


Fig 5.1.1 Normal State

The results in Figures 3 and 4 showcase the Real-Time Crowd Detection system's functionality, recognizing people through green rectangular boxes and actively monitoring their count. Pre-recorded people count demonstrates the system's ability to display the number of individuals within the monitored area. The green boxes affirm the effectiveness of the crowd detection algorithm. Utilizing a threshold value, the system triggers on-screen text messages when the people count exceeds, indicating overcrowding, and employs pyttsx3 for voice alerts. Conversely, when the count is below the threshold, a normal state message is displayed. This dynamic system responds effectively to varying crowd densities, offering a comprehensive approach to crowd management with visual, on-screen, and voice alerts. These results set the stage for enhanced safety and crowd control measures in diverse environments.

Figure 5 demonstrates the Real-Time Crowd Detection system's live video features, presenting insights on people count and response mechanisms. The real-time counting of individuals in the live video feed provides immediate and dynamic crowd density information. Human recognition is visually highlighted through green rectangular boxes, showcasing the system's proficiency in identifying people. Utilizing a threshold for live people count, the system triggers onscreen text messages and voice alerts via pyttsx3 upon surpassing the threshold, indicating overcrowding. Voice messages complement on-screen alerts, efficiently notifying users or authorities. When the people count is below the threshold, a normal state message reassures that the monitored area is stable. This real-time responsiveness, coupled with visual, onscreen, and voice alerts, contributes to a comprehensive crowd management approach. The integration of live video analysis further enhances the system's effectiveness, establishing a robust foundation for improved safety and control measures in real-world environments.

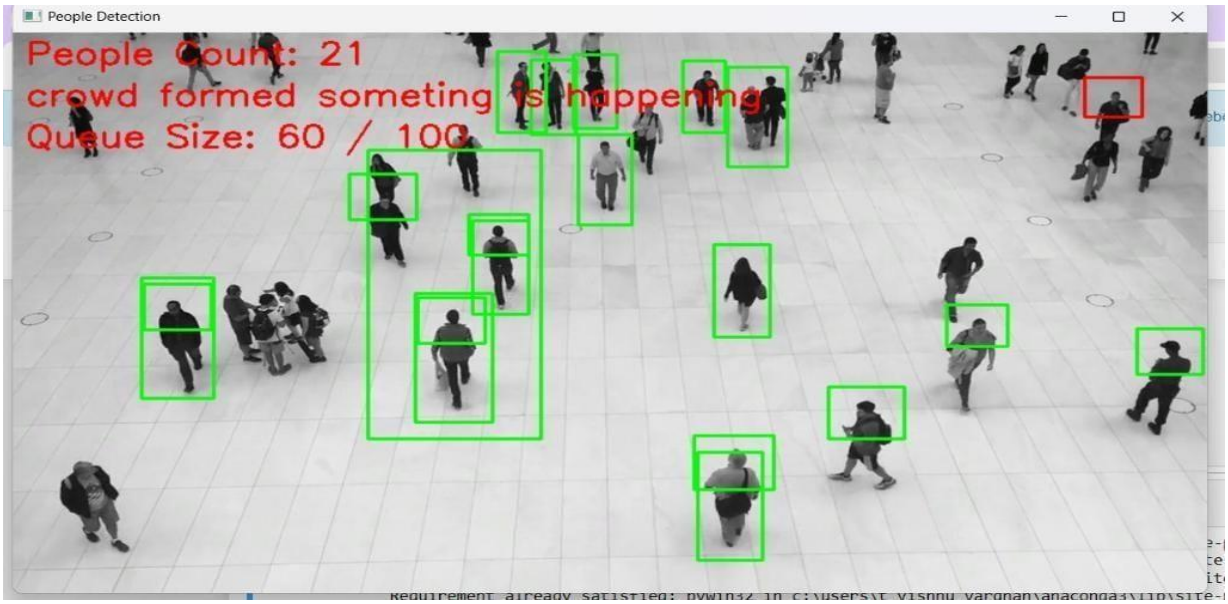


Fig 5.1.2 Crowd Formed

5.2 SAMPLE CODE

```
#!/usr/bin/env python
# coding: utf-8

# In[2]:


import cv2
import winsound

# from gtts import gTTS
# import os
import pyttsx3

# ...

engine = pyttsx3.init()

engine.setProperty('rate', 100)

pedestrian_cascade = cv2.CascadeClassifier('haarcascade_fullbody (1).xml')
fullbody_cascade = cv2.CascadeClassifier('haarcascade_upperbody.xml')

video_source = "pep.mp4"
cap = cv2.VideoCapture(video_source)

people_count = 0
group_count = 0
group_threshold = 5
message = ""
queue = []
max_queue_size = 100
beep_played = False

while True:
    ret, frame = cap.read()
```

```
    if not ret:
        break

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    pedestrians = pedestrian_cascade.detectMultiScale(gray, scaleFactor=1.1,
minNeighbors=5, minSize=(30, 30))

    fullbodies = fullbody_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5,
minSize=(30, 30))

    all_people = list(pedestrians) + list(fullbodies)

    frame_group_count = 0

    for (x, y, w, h) in all_people:
        if w * h > 1000:
            if frame_group_count == 0:
                group_count += 1
            frame_group_count += 1
            color = (0, 0, 255) if frame_group_count > group_threshold else (0, 255, 0)
            cv2.rectangle(frame, (x, y), (x + w, y + h), color, 2)

            queue_area = (x > 300 and x < 500 and y > 200 and y < 400)

            if queue_area:
                queue.append((x, y, w, h))

    queue = [(x, y, w, h) for (x, y, w, h) in queue if x > 300]

    people_count = len(all_people)
    if frame_group_count > group_threshold:
        message = "crowd formed something is happening"

        if not beep_played and cap.get(cv2.CAP_PROP_POS_MSEC) >= 1000:
            engine.say(message)
```

```
            engine.runAndWait()

            beep_played = True
    elif frame_group_count < group_threshold:
        message = "The Area is in normal state"
    else:
        message = ""

    queue_status = f"Queue Size: {len(queue)} / {max_queue_size}"
    cv2.putText(frame, queue_status, (10, 110), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0,
255), 2)

    cv2.putText(frame, f"People Count: {people_count}", (10, 30),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2)
    cv2.putText(frame, message, (10, 70), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255),
2)

    cv2.imshow("People Detection", frame)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()

# In[3]:




# In[ ]:


# In[ ]:


# In[ ]:
```

In[ ]:

In[ ]:

In[ ]:

In[ ]:

In[ ]:

In[ ]:

In[ ]:

In[ ]:

In[ ]:

In[ ]:

In[ ]:

6.SYSTEM TESTING

Limitation Alerting System

System testing ensures the system's functionality, reliability, and performance in real-world conditions.

1. Functional Testing

Verified video capture, crowd detection, threshold analysis, and audio alert generation.

Ensured Haar Cascade classifiers accurately detect individuals and trigger alerts when thresholds are exceeded.

2. Integration Testing

Tested seamless interaction between video feed, detection module, and alert generation.

3. Performance Testing

Measured real-time frame processing speed and system scalability.

Evaluated detection accuracy under different lighting and environmental conditions.

4. Stress Testing

Simulated high crowd density and extended usage to assess stability.

5. User Acceptance Testing

Tested customizable thresholds and alert effectiveness with user feedback.

6. Results

The system successfully met objectives, with minor detection challenges in low light or occlusion scenarios. It is reliable and ready for deployment.

7.SCREENSHOTS

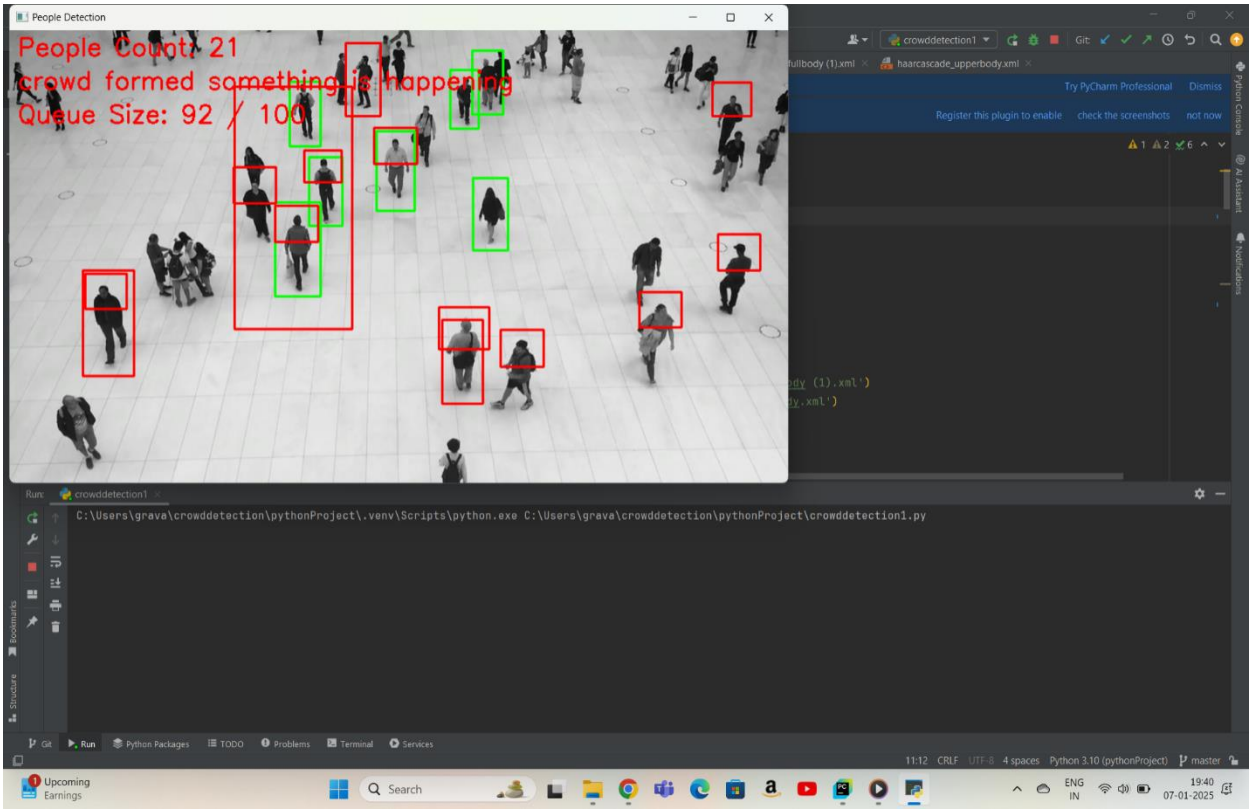


Fig:7.1Crowd formed detected by Media

Fig:7.2 Alert in text form

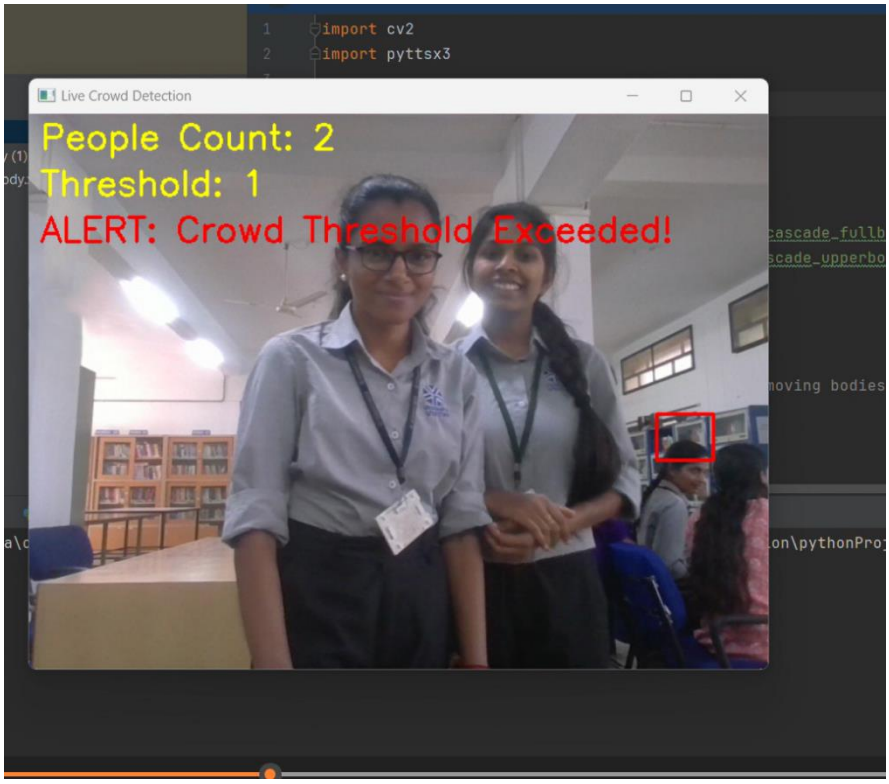


Fig:7.3Crowd formed detected by Media

Fig:7.4Area in normal state

8.CONCLUSION

The crowd limitation alerting system, utilizing Pyttsx, Haar Cascade files, and OpenCV, is an efficient and cost-effective solution for monitoring and managing crowd density. By leveraging Haar Cascade-based detection, the system can identify and track people in real time, while Pyttsx generates voice alerts to warn when crowd limits are exceeded. Its reliance on open-source libraries makes it affordable and accessible, and its automation reduces the need for manual intervention, making it ideal for environments such as public spaces, events, or workplaces.

However, the system faces challenges such as reduced accuracy in complex or dynamic environments, where factors like poor lighting or overlapping objects can lead to false positives or negatives. While Haar Cascade classifiers are effective, their pre-trained nature limits customization for specific scenarios. To improve performance, integrating advanced deep learning models like YOLO or MobileNet-SSD and incorporating multi-modal sensors such as thermal cameras could enhance detection accuracy and reliability, ensuring a more robust and scalable solution for crowd management.

9.FUTURE SCOPE

The proposed system proves effective in dynamically monitoring crowds. The threshold- based alert system, complemented by visual indicators and voice alerts, ensures prompt response to potential overcrowding. The system's accurate human recognition and real-time adaptability make it a valuable tool for crowd management. With the potential for further enhancements, it lays a strong foundation for improving safety measures in diverse environments. Overall, the integration of computer vision and audio notifications demonstrates practical applications for real- world crowd control.

The future scope of Real-Time Crowd Detection using OpenCV and pyttsx3 holds promising avenues for advancement. Integration with cutting-edge technologies such as edge computing and IoT will enhance real-time processing and data collection capabilities, providing a more comprehensive understanding of crowd dynamics. Further development of advanced machine learning models can empower the system to predict and proactively manage crowd behavior, ensuring a more intelligent approach to overcrowding management. Expanding the capabilities of the mobile application, with features like real-time alerts and collaborative reporting, will foster greater user engagement and contribute to a more interactive crowd management ecosystem. Additionally, incorporating geospatial mapping for spatial visualization and continuous optimization for performance will solidify the system's position at the forefront of innovative solutions, contributing to improved safety and efficiency in managing crowded environments.

10.BIBLIOGRAPHY

10.1 REFERENCES:

1. CG Abhinu, P Aswin, Kiran Krishnan, Bonymol Baby, and KS Angel Viji. Multiple object tracking using deep learning with yolo v5. International Journal of Engineering Research & Technology, 9(13):4751, 2021.
2. Laith Alzubaidi, Jinglan Zhang, Amjad J Humaidi, Ayad Al-Dujaili, Ye Duan, Omran Al-Shamma, Jose Santamar ´ ´ıa, Mohammed A Fadhel, Muthana Al-Amidie, and Laith Farhan. Review of deep learning: Concepts, cnn architectures, challenges, applications, future directions. Journal of big Data, 8:1–74, 2021.
3. Md Israfil Ansari and Jaechang Shim. People counting system using raspberry pi. Journal of Multimedia Information System, 4(4):239–242, 2017.
4. Michael D Breitenstein, Fabian Reichlin, Bastian Leibe, Esther KollerMeier, and Luc Van Gool. Online multiperson tracking-by-detection from a single, uncalibrated camera. IEEE transactions on pattern analysis and machine intelligence, 33(9):1820–1833, 2010.
5. Mayur D Chaudhari and Archana S Ghotkar. A study on crowd detection and density analysis for safety control. International journal of computer sciences and engineering, 6:424–428, 2018.
6. Lijia Deng, Shui-Hua Wang, and Yu-Dong Zhang. Fully optimized convolutional neural network based on small-scale crowd. In 2020 IEEE International Symposium on Circuits and Systems (ISCAS), pages 1–5. IEEE, 2020.
7. Zizhu Fan, Hong Zhang, Zheng Zhang, Guangming Lu, Yudong Zhang, and Yaowei Wang. A survey of crowd counting and density estimation based on convolutional neural network. Neurocomputing, 472:224–251, 2022.
8. Min Fu, Pei Xu, Xudong Li, Qihe Liu, Mao Ye, and Ce Zhu. Fast crowd density estimation with convolutional neural networks. Engineering Applications of Artificial Intelligence, 43:81–88, 2015.
9. Haroon Idrees, Imran Saleemi, Cody Seibert, and Mubarak Shah. Multisource multi-scale counting in extremely dense crowd images. In Proceedings of the

IEEE conference on computer vision and pattern recognition, pages 2547–2554, 2013.

10. Hasith Karunasekera, Han Wang, and Handuo Zhang. Multiple object tracking with attention to appearance, structure, motion and size. IEEE Access, 7:104423–104434, 2019.
11. Min Li, Zhaoxiang Zhang, Kaiqi Huang, and Tieniu Tan. Estimating the number of people in crowded scenes by mid based foreground segmentation and head-shoulder detection. In 2008 19th international conference on pattern recognition, pages 1–4. IEEE, 2008.
12. Songyan Ma and Tiancang Du. Improved adaboost face detection. In 2010 International Conference on Measuring Technology and Mechatronics Automation, volume 2, pages 434–437. IEEE, 2010.
13. Yunqi Miao, Jungong Han, Yongsheng Gao, and Baochang Zhang. Stcnn: Spatial- temporal convolutional neural network for crowd counting in videos. Pattern Recognition Letters, 125:113–118, 2019.
14. Davidson Kamala Dhas Milton and Arun Ra Velraj. Crowd size estimation and detecting social distancing using raspberry pi and opencv. International Journal of Electronics and Telecommunications, 69, 2023.
15. Viet-Quoc Pham, Tatsuo Kozakaya, Osamu Yamaguchi, and Ryuzo Okada. Count forest: Co-voting uncertain number of targets using random forest for crowd density estimation. In Proceedings of the IEEE international conference on computer vision, pages 3253– 3261, 2015.
16. Chong Shang, Haizhou Ai, and Bo Bai. End-to-end crowd counting via joint learning local and global count. In 2016 IEEE International Conference on Image Processing (ICIP), pages 1215–1219. IEEE, 2016.
17. Li Tan, Xu Dong, Yuxi Ma, and Chongchong Yu. A multiple object tracking algorithm based on yolo detection. In 2018 11th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), pages 1–5. IEEE, 2018.
18. Qi Wang, Junyu Gao, Wei Lin, and Xuelong Li. Nwpu-crowd: A largescale benchmark for crowd counting and localization. IEEE transactions on pattern

analysis and machine intelligence, 43(6):2141–2149, 2020.

19. Qi Wang, Junyu Gao, Wei Lin, and Yuan Yuan. Learning from synthetic data for crowd counting in the wild. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 8198– 8207, 2019.
20. Yi Wang, Junhui Hou, Xinyu Hou, and Lap-Pui Chau. A self-training approach for point- supervised object detection and counting in crowds. IEEE Transactions on Image Processing, 30:2876–2887, 2021.
21. Yi Wang, Xinyu Hou, and Lap-Pui Chau. Dense point prediction: A simple baseline for crowd counting and localization. In 2021 IEEE International Conference on Multimedia & Expo Workshops (ICMEW), pages 6.EE, 2021.
22. Xing Wei, Yuanrui Kang, Jihao Yang, Yunfeng Qiu, Dahu Shi, Wenming Tan, and Yihong Gong. Scene-adaptive attention network for crowd counting. arXiv preprint arXiv:2112.15509, 2021.
23. Shaokai Wu and Fengyu Yang. Boosting detection in crowd analysis via underutilized output features. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 15609– 15618, 2023.
24. Mingliang Xu, Zhaoyang Ge, Xiaoheng Jiang, Gaoge Cui, Pei Lv, Bing Zhou, and Changsheng Xu. Depth information guided crowd counting for complex crowd scenes. Pattern Recognition Letters, 125:563–569, 2019.
25. Binyu Zhang, Yunhao Du, Yanyun Zhao, Junfeng Wan, and Zhihang Tong. I-mmccn: Improved mmccn for rgb-t crowd counting of drone images. In 2021 7th IEEE International Conference on Network Intelligence and Digital Content (IC-NIDC), pages 117–121. IEEE, 2021.