

Swacch Suraksha - AI Enabled Smart Surveillance System for Anti - Littering Enforcement

Harshath B S¹, Komala K², Abhiram L N³, G Seema⁴, K H Surabhi⁵

^{1,2,3,4,5}Dept. of Electronics & Communication Engineering, Sri Siddhartha Institute of Technology, SSAHE, Tumakur, India

komalak@ssit.edu.in, harshathsuresh50@gmail.com, samurel974@gmail.com, seemagseema8@gmail.com, khsurabhi616@gmail.com

Abstract - This paper presents an intelligent real-time surveillance system for automatic detection and penalization of individuals who discard objects (e.g., bags, cups, and bottles) beyond a defined safety boundary. The system integrates YOLOv5-based object detection with a lightweight tracking mechanism to monitor object trajectories and identify boundary violations. For user identification, LBPH-based face recognition is employed, providing robustness under varying illumination and partial occlusions. A face enrollment module generates a local dataset to train a persistent recognition model, while a short-term memory mechanism ensures stable user association during live monitoring.

Detected violations trigger automated alerts via the Twilio API, transmitting user identity, object type, and penalty details to registered mobile numbers. A Flask-based web interface enables secure user management, face training, and real-time video monitoring. System data is maintained using an SQLite database to ensure reliability and portability. The proposed system provides a fully automated and scalable solution for enforcement in campuses, industrial environments, public spaces, and smart city applications.

Keywords: *Intelligent Surveillance System (ISS), YOLOv5 Object Detection(YOLOv5), Local Binary Patterns Histogram (LBPH), Computer Vision (CV), Flask Web Frame Work (FLASK), Application Programming Interface (API), Short Message Service (SMS) Alert System (SMS).*

1. Introduction

The rapid growth of artificial intelligence and embedded computer vision technologies has significantly transformed modern surveillance systems, shifting them from basic video monitoring solutions into intelligent, context-aware, and autonomous decision-making platforms. Traditional CCTV systems merely record video footage, placing the burden of interpretation, monitoring, and response entirely on human operators, which often leads to delays, oversight, and inefficiencies, especially in high-traffic or highly dynamic environments. To address these limitations, modern surveillance applications increasingly incorporate machine learning, real-time object detection, face recognition, automated tracking, and cloud-based alert systems. This project is an advanced implementation of such next-generation surveillance, designed to detect and prevent undesirable behaviors specifically the act of throwing or dropping objects such as bags, cups, and bottles - within monitored areas. The system integrates

real-time object detection (YOLOv5), robust face recognition (LBPH), lightweight object tracking, automated violation analysis, and Twilio-based SMS alerts, all deployed through an interactive Flask web-based interface. The motivation behind this work emerges from growing concerns over public safety, waste mismanagement, and enforcement inefficiencies in environments such as college campuses, industrial workplaces, public pathways, commercial buildings, transportation terminals, and restricted premises. Despite the availability of surveillance cameras, violations like littering, unauthorized object disposal, or harmful actions often go unnoticed due to the lack of automated monitoring, human fatigue, and the sheer scale of areas under observation. Therefore, a fully automated, intelligent, real-time monitoring solution becomes essential for ensuring cleanliness, discipline, accountability, and safety across public and private environments. The rise of deep learning frameworks has revolutionized object detection, enabling models such as YOLOv5 to recognize multiple classes—including bags,

cups, and bottles—with high frame rates and impressive accuracy, even on modest computational hardware. YOLOv5's one-stage detection architecture and optimized inference pipeline make it particularly suitable for real-time applications where rapid response is crucial. In this project, YOLOv5 is employed to detect specified objects continuously while a bounding-box tracker monitors their motion and determines whether an item has been thrown below a defined ground-threshold line. This threshold-based rule serves as the behavioral logic for identifying violations, allowing the system to classify actions not simply by detecting an object but by interpreting its motion relative to the environment. In parallel, the system incorporates an identity-verification module using the Local Binary Patterns Histograms (LBPH) method, a classical but extremely reliable approach for face recognition under varied lighting and real-time constraints. Users must first undergo face enrollment, where multiple samples of their facial images are captured and stored. This dataset is used to train an LBPH classifier, enabling precise identification of individuals appearing in the camera feed. The face-recognition module includes a temporal memory buffer, allowing the system to maintain user identity for several seconds even if the face is momentarily occluded, blurred, or out of frame. This ensures that any detected violation is accurately linked to the responsible individual without ambiguity or manual intervention. To transform the raw detection and recognition capabilities into a practical enforcement system, this project integrates a cloud communication layer using the Twilio API, enabling instant **SMS notifications** whenever a violation occurs. If the system recognizes a registered user and detects an object being thrown beyond the ground level, an automated alert is sent to the user's phone number, containing the name, object detected, and penalty details. This ensures immediate accountability while creating a strong deterrent effect against unwanted actions. The entire solution is wrapped within a Flask framework, providing user registration, login authentication, live video streaming, face training, and surveillance dashboards through a browser-friendly interface. SQLite serves as the lightweight relational database backend, handling user records, face data references, and event logs with reliability and minimal overhead. The project emphasizes modularity, scalability, and ease of deployment: it can operate on standard computing hardware with a single camera and can be extended to support multiple cameras, additional object classes, or advanced behavioral rules. Furthermore, the architecture allows integration with cloud storage, centralized

monitoring systems, administrative reporting dashboards, and automated fine-management systems, making it adaptable to future expansions. Overall, this project demonstrates a comprehensive integration of artificial intelligence, real-time computer vision, object analytics, identity verification, and automated communication to create a fully autonomous smart surveillance system. It transcends conventional monitoring by not only detecting objects but also analyzing human behavior, identifying offenders, and issuing automated responses without manual oversight. Such capabilities position the system as a powerful tool for smart-city development, campus discipline management, industrial safety enforcement, and public-space monitoring.

2. Literature Survey

The evolution of intelligent surveillance systems has been shaped by rapid advancements in computer vision, deep learning, and embedded AI technologies, enabling automatic detection of human actions, object interactions, and behavioral violations in real-time environments. Early surveillance mechanisms relied primarily on motion detection, background subtraction, and handcrafted feature extraction methods such as HOG, SIFT, and Haar-like descriptors, but these approaches exhibited poor performance under varying illumination, occlusions, and dynamic backgrounds. With the emergence of convolutional neural networks (CNNs), object detection frameworks such as R-CNN, Fast R-CNN, and Faster R-CNN introduced region-proposal architectures capable of greatly improving detection accuracy, although their computational complexity made them unsuitable for real-time applications. The paradigm shifted significantly with the introduction of the YOLO (You Only Look Once) family of models, which transformed object detection into a single-stage, unified prediction problem, achieving high accuracy while sustaining real-time performance. Recent versions such as YOLOv5 have become widely used in academic and industrial applications due to their balance of speed, model size, and practical deployability on CPU-based systems, making them suitable for real-time detection of small objects like bags, cups, and bottles in live surveillance feeds. In parallel, facial recognition research progressed from traditional PCA and LDA methods toward more robust feature-based approaches like LBPH (Local Binary Patterns Histograms), which remains a popular technique for real-time face recognition due to its resilience to lighting variations, low computational cost, and compatibility with standard CCTV camera

resolutions. Although deep-learning-based models such as FaceNet and ArcFace offer higher accuracy, their heavy computational requirements make LBPH preferable for systems requiring lightweight, CPU-based real-time inference, particularly in embedded or resource-limited deployments.

In the domain of tracking and behavior analysis, literature highlights significant progress through algorithms such as SORT and DeepSORT, which introduced Kalman filtering and deep metric learning to maintain persistent object identities across frames. However, their dependency on deep re-identification networks often increases system complexity. As a result, many practical systems employ simplified IoU-based tracking mechanisms for single-camera environments, focusing instead on rule-based behavioral analysis such as trajectory monitoring, region-based alert generation, and threshold-based event detection. Studies on litter detection and environmental monitoring emphasize the necessity of interpreting contextual interactions rather than merely detecting objects. Research exploring intelligent waste monitoring systems demonstrates that simply identifying objects like bottles or cups is insufficient; instead, behavior-driven logic - such as determining whether an object falls below a spatial threshold or exhibits a downward trajectory - is required to classify littering events accurately. Additionally, the literature on identity-linked surveillance stresses the importance of combining object detection with face recognition to assign accountability, reduce anonymity, and support automated enforcement frameworks. Furthermore, several studies examine the integration of cloud-based communication platforms, such as SMS APIs or IoT gateways, to facilitate real-time alerts and notifications, thereby enhancing incident response in safety-critical applications. The convergence of these technologies-fast object detection, robust face recognition, lightweight tracking, and automated messaging forms the foundation for modern intelligent surveillance solutions capable of autonomous monitoring and disciplinary decision-making.

Overall, existing research establishes a clear trajectory toward fully automated, AI-driven surveillance systems that not only detect objects but also understand actions, identify individuals, and autonomously trigger responses. However, the literature shows that most prior implementations focus separately on object detection or face recognition, with limited work combining both into a unified real-time enforcement system. This gap

highlights the need for integrated solutions capable of detecting specific objects (such as bags, cups, and bottles), analyzing object trajectories, identifying responsible individuals using real-time facial recognition, and issuing immediate alerts through automated communication channels all of which form the core innovation of the present project.

3. Proposed Methodology

The methodology adopted for this project is a multi-stage integration of computer vision, machine learning, identity verification, real-time tracking, and automated communication, structured to deliver a fully autonomous surveillance and violation detection system capable of monitoring the disposal of objects such as bags, cups, and bottles. The process begins with the preparation and configuration of the development environment, where the Flask framework, OpenCV, YOLOv5, SQLite, and Twilio APIs are installed and interconnected to form the foundational software stack. The methodology first focuses on face dataset acquisition, where users perform a face registration process using a webcam connected to the system. During this stage, 80 high-quality grayscale facial images are captured per user under varying angles and lighting conditions. These images undergo preprocessing, including histogram equalization and resizing to a standard 200×200 resolution, ensuring consistent LBPH model training.

Once the dataset is prepared, the LBPH algorithm extracts local binary texture patterns from each facial region and computes histograms that represent unique facial signatures. These histograms are then used to train the LBPH face recognizer, producing a face model capable of recognizing registered individuals in real time with high confidence and robustness to variations such as illumination changes and partial occlusions. The trained model is saved locally and updated automatically when new users enroll, allowing continuous scalability of the identity database. The second stage of the methodology involves the object detection pipeline, where YOLOv5 is employed to detect target objects - bags, cups, and bottles in live video frames. YOLOv5 is selected for its high speed, accuracy, and suitability for CPU-based execution, making it practical for real-time surveillance without requiring GPU hardware. Each frame captured from the webcam undergoes preprocessing via the letterbox resizing technique to preserve aspect ratio while preparing the image for YOLOv5 inference.

The model predicts bounding boxes, confidence scores, and class labels, following which Non-Maximum Suppression (NMS) filters out redundant detections. The system continuously monitors the spatial coordinates of each detected object and maps them onto the frame. To analyze object motion across time, the methodology incorporates a lightweight IoU-based tracking algorithm that maintains track IDs for each object across consecutive frames. Instead of computationally expensive deep re-identification strategies, the chosen approach compares bounding box overlaps to determine object continuity. This makes the tracker simple, efficient, and well-suited for single-camera environments. The tracker also stores each object's state, label, and loss counter, enabling it to manage frame-to-frame inconsistencies while maintaining high reliability.

The third major component of the methodology concerns behavioral rule-based violation detection, which is essential for identifying the act of throwing or dropping objects below a predefined ground-level threshold. A virtual “ground line” is drawn at a specific ratio of the frame height, typically around 80% of the vertical dimension. For any tracked object, its center point is computed from the bounding box coordinates. When this center crosses below the ground threshold, the system interprets the behavior as a disposal action. This threshold-based violation logic ensures accurate classification of littering events without relying on complex gesture recognition or motion analysis networks. The moment a violation is detected, the system checks whether a valid face has been recognized within the last few seconds using a face-memory buffer mechanism. This buffer stores the latest recognized face ID and timestamp, allowing the system to maintain identity continuity even when the face disappears from the frame due to motion, temporary occlusion, or orientation changes. By combining object trajectory analysis with short-term identity preservation, the system guarantees reliable linking of violation events to individuals. The fourth stage of the methodology involves the automated SMS notification module, which uses the Twilio API to communicate violation alerts. If an object is detected below ground level and a recognized user is associated with the event, the system automatically composes and sends an SMS message containing the user's name, the detected object type, and penalty information. This occurs instantly and without human intervention, ensuring prompt accountability.

The Twilio API is integrated securely within the application, and message logs are maintained for verification and auditing. The final component of the methodology focuses on the web application layer, implemented using Flask to provide an interactive dashboard for administrators. This web interface includes modules for user registration, password-protected login, face training initiation, and live surveillance video streaming through MJPEG feeds. Flask manages all HTTP requests, session handling, and template rendering, while SQLite serves as the lightweight database for storing credentials, face-user mappings, and event-related details. The system's architecture emphasizes modularity, allowing each component—face recognition, object detection, tracking, violation logic, and SMS notification—to operate independently yet harmoniously through well-defined interfaces. This modular design also supports future scalability, such as expanding to multiple camera feeds, integrating cloud storage, enhancing the detection model with more classes, or upgrading face recognition to deeper neural architectures.

4. Block Diagram

The block diagram of the proposed system illustrates the overall architecture and functional flow of the intelligent surveillance framework. It consists of multiple interconnected modules, including video acquisition, object detection using YOLOv5, object tracking and violation analysis, face recognition using LBPH, and an alert generation unit. The input is captured through a camera, where real-time frames are processed to detect specified objects such as bags, cups, and bottles. The tracking module monitors object movement to determine whether it crosses a predefined safety boundary.

Simultaneously, the face recognition module identifies individuals using a pre-trained LBPH model, enabling association between detected actions and registered users. Upon detecting a violation, the system triggers an automated alert via an API-based messaging service and logs the event in the database. A web-based interface supports system monitoring, user management, and live video streaming. The block diagram provides a clear representation of data flow and interaction among these components, ensuring efficient and automated system operation.



1. Object Detection(YOLOv5)

The YOLOv5 Object Detection module represents the first major processing stage of the system and is responsible for detecting objects such as bags, cups, and bottles directly from the live video feed. When each frame enters the pipeline, it undergoes preprocessing through the letterbox resizing algorithm, which ensures that the input maintains its aspect ratio while fitting YOLOv5's expected image dimensions.

This step minimizes distortion and enhances detection precision. The frame is then normalized, converted into tensor format, and passed into the YOLOv5 model, which evaluates it in a single forward pass using its advanced CSPDarknet-based backbone and PANet-based feature aggregation network. YOLOv5 produces outputs consisting of bounding-box coordinates, class probabilities, and confidence scores for every potential object detected in the frame. These raw outputs are subsequently filtered using Non-Maximum Suppression (NMS) to eliminate duplicate predictions and retain only the most accurate detections. YOLOv5's ability to operate efficiently on CPU hardware allows the system to maintain real-time performance without requiring expensive GPUs, making it ideal for surveillance environments. The selected object classes—bag, cup, and bottle—are central to the project's aim of detecting littering-related behaviors.

Each detection is mapped back to the frame's original resolution, enabling precise spatial referencing for downstream tracking and violation analysis. The YOLOv5 module ensures that objects are consistently and reliably identified across varying lighting conditions, camera angles, and environmental scenarios. Its robustness enables the system to perform reliably in live, dynamic environments where objects may appear, move quickly, or partially occlude. The outputs from this module form the foundation for the tracking and behavioral logic modules that follow.

2. Face Recognition (LBPH)

The Face Recognition module, powered by the Local Binary Patterns Histograms (LBPH) algorithm, provides the essential identity verification capability that binds each violation to a specific user. This component begins with the face dataset captured during the registration phase, where multiple grayscale images are collected for each individual. LBPH processes each face by dividing it into small local regions and computing binary patterns that describe texture transitions around each pixel. These patterns are aggregated into histograms, forming a unique facial signature for every user. During real-time inference, every incoming frame is converted to grayscale, equalized, and scanned using the Haar Cascade classifier to detect facial regions. Detected faces are resized and normalized before being passed to the LBPH recognizer, which compares their histogram signatures with those stored in the trained model. The system outputs the user ID along with a confidence score indicating the closeness of the match. A threshold mechanism determines whether the detected face is reliable enough to consider as an authenticated identity. To enhance stability, a face-memory buffer stores the last recognized user ID and its timestamp, allowing the system to maintain identity continuity even when the face momentarily disappears due to occlusion or head movement. This design ensures that a violation occurring a few seconds after recognition is still attributed to the correct individual. LBPH is chosen due to its low computational overhead, making it suitable for real-time applications without dependency on deep-learning GPUs. It is robust to changes in lighting, partial facial variations, and moderate pose differences. By linking object detection events with recognized faces, the

module enables automatic enforcement and accountability, forming a critical component of the surveillance system's effectiveness.

3. Tracking & Violation Logic

The Tracking & Violation Logic module is responsible for continuously monitoring detected objects across consecutive frames and determining whether they cross the predefined ground threshold, indicating a littering violation. This module begins by taking the bounding boxes produced by YOLOv5 and associating them with existing object tracks using IoU (Intersection over Union), which calculates the overlap between current detections and previously tracked objects. If a detection sufficiently overlaps with a stored track, the system updates that track's position; otherwise, it initializes a new track with a unique ID. This mechanism ensures that objects such as bags, cups, and bottles are consistently followed throughout their motion. The module calculates the centroid of each bounding box and compares the vertical coordinate against a virtual ground line, typically located at 80% of the frame height. If the centroid falls below this line, the system interprets this behavior as an act of dropping or throwing the object. Additional logic prevents false triggers by requiring the object to cross the threshold decisively rather than due to noise or momentary camera movement. The module also monitors track disappearance: when an object temporarily vanishes, its track remains active for a limited number of frames (defined by MAX_LOST), ensuring stability despite occlusions. The violation decision-making process is tightly integrated with the face-recognition memory: a violation is only confirmed when a recognized user has appeared shortly before the event. This ensures that the responsibility is accurately attributed. The module sets the "thrown" flag once a violation is confirmed, preventing duplicate alerts while maintaining a clear record of the incident. As the central analytical engine, this block transforms raw detections into meaningful behavioral insights, enabling automated enforcement.

4. Violation Alert Module (SMS/Twilio)

The Violation Alert Module is responsible for delivering immediate SMS notifications to the individual responsible for a detected violation. When the Tracking & Violation Logic concludes that an object—such as a bag, cup, or bottle—has been thrown below the ground-level threshold, this module retrieves the recognized user's name and registered phone number and generates

a descriptive, structured alert message. The system integrates the Twilio REST API, which allows secure, reliable SMS delivery across global networks. The module constructs a message containing the user's name, the object detected, the nature of the violation, and the penalty amount (₹500).

The API authentication uses the account SID and Auth Token to ensure that only authorized requests are made. Upon sending, the system receives a unique message SID as confirmation, which can be logged for auditing. Error-handling routines detect issues such as network delays, invalid phone numbers, or SMS gateway failures and print diagnostic information to ensure reliability. The module is designed to operate instantly the moment a violation occurs, ensuring real-time feedback and strengthening accountability. Automation eliminates dependence on manual monitoring staff and enforces discipline consistently. It also scales to large environments because SMS delivery is handled by cloud infrastructure rather than local services. This module's design ensures compliance, transparency, and deterrence by combining identity recognition with automated enforcement. Through its integration with the system's recognition and detection modules, the alert mechanism becomes a powerful tool for creating behavior change in monitored environments.

5. Flask Web Dashboard & Database Layer

The Flask Web Dashboard and Database Layer constitute the user-facing control panel and the core data management mechanism of the system. Through the Flask framework, administrators can perform secure login authentication, register new users, initiate face dataset capturing, and monitor the live camera feed. The dashboard provides buttons and navigation routes that correspond to back-end logic for training the LBPH model, starting object detection, and supervising system performance. Flask handles routing through HTTP endpoints such as /login, /dashboard, /train_face, and /video, each rendering a corresponding HTML template. The real-time video stream is implemented using an MJPEG response generator, sending encoded JPEG frames in a continuous stream to the browser. The back-end communicates seamlessly with the SQLite database, which stores essential information such as usernames, hashed passwords, phone numbers, and mappings between face-user IDs. SQLite is chosen for its reliability, lightweight architecture, and minimal configuration requirements, making it ideal for

embedded surveillance solutions. The database tables—users and face_users—ensure proper separation between login credentials and face recognition records. Any violation event can also be logged for administrative review or reporting. Flask sessions maintain user authentication and ensure secure access to the dashboard. Through this web interface, non-technical staff can operate the system without needing direct access to underlying code. The dashboard's design emphasizes clarity, stability, and real-time responsiveness, allowing administrators to track violations, confirm alerts, and manage user data efficiently. Together, the Flask and SQLite layers provide the foundational backbone connecting all modules into a cohesive, user-friendly surveillance ecosystem.

The System is composed of following Components:

1. Raspberry Pi Module 5



The Raspberry Pi is a low-cost, credit-card-sized single-board computer designed to promote computer education, rapid prototyping, and embedded system development. Developed by the Raspberry Pi Foundation in the UK, it has evolved into one of the most popular platforms for electronics, robotics, IoT, AI, and automation projects due to its compact size, powerful hardware, and support for a wide range of programming languages. Although small, the Raspberry Pi functions like a full desktop computer and can run a complete operating system such as Raspberry Pi OS (Linux-based), Ubuntu, or even Windows IoT Core. It supports essential components such as USB ports, HDMI output, Wi-Fi, Bluetooth, GPIO pins, camera interface, and audio output, making it highly versatile for both academic and industrial applications.

2. Raspberry Pi Camera Module 3



The Raspberry Pi Camera Module 3, commonly known as the Pi Camera, is a compact, high-resolution imaging device specifically designed for use with Raspberry Pi boards. It connects directly to the Raspberry Pi through the CSI (Camera Serial Interface) port, enabling high-speed image and video capture with minimal latency. Because of its small size, low power consumption, and ability to integrate seamlessly with deep learning and computer vision libraries such as OpenCV, TensorFlow, and YOLO, the Pi Camera is widely used in robotics, surveillance, agriculture automation, and machine vision applications.

The camera module is available in different versions such as V1 (5 MP OV5647 sensor), V2 (8 MP Sony IMX219 sensor), and High Quality Camera (12.3 MP Sony IMX477). The V2 camera is the most commonly used due to its balance between price and performance. It supports still images up to 3280×2464 resolution, 1080p video at 30fps, 720p at 60fps, and 480p at 90fps, making it suitable for both photography and real-time video processing tasks. The latest HQ Camera even supports interchangeable lenses, enabling advanced imaging for industrial and scientific applications.

One of the major advantages of the Pi Camera is its plug-and-play compatibility. With just a ribbon cable connected to the CSI port, the camera can be programmed using Python libraries like Picamera2, which offers a simple interface for capturing images, recording videos, and streaming frames for object detection.

It also integrates perfectly with OpenCV, allowing users to perform tasks such as face recognition, weed detection in agriculture, gesture tracking, traffic monitoring, and smart surveillance. The Pi Camera is especially beneficial for embedded systems because it consumes very little power while providing high-quality outputs. It is also supported across all Raspberry Pi OS versions, enabling real-time image acquisition without requiring external USB webcams. This makes it ideal for building smart IoT devices, autonomous robots, and machine

learning projects where size, speed, and processing efficiency are critical

Software Requirements

Operating System

- Windows 10 / 11
- Linux (Ubuntu 20.04+)
- macOS (optional, but compatible)

Programming Language

- Python 3.8+

Required Python Libraries

- Flask (for web dashboard)
- OpenCV (for face detection, image preprocessing, camera handling)
- torch / torchvision (PyTorch backend for YOLOv5)
- numpy (matrix operations)
- sqlite3 (database driver)
- twilio (SMS alert integration)
- werkzeug (password hashing and security)

Frameworks / Models

- YOLOv5 object detection model
- LBPH Face Recognizer (OpenCV's face module)
- Haar Cascade classifier for face detection

3. Functional Requirements

Object Detection

- System must detect bag, cup, bottle objects in real-time.
- YOLOv5 must return bounding boxes, label names, and confidence scores.

Face Recognition

- System must capture at least 40–80 face samples per user.
- Must identify users via LBPH with a defined confidence threshold.

Tracking

- Must track detected objects frame-by-frame using IoU-based tracking.

- Must maintain object IDs during movement.

Violation Detection

- Must identify when an object crosses the ground-level threshold.
- Must trigger violation only once per object lifecycle.

Alert System

- On violation, system must:
 - Fetch user name + phone number
 - Compose SMS alert
 - Send message via Twilio API instantly

Web Application

- Must support:
 - User registration
 - Login authentication
 - Face training
 - Live video monitoring

Database

- Must store:
 - User login details (username, hashed password)
 - Face user details (name, phone number)
 - Face model file and dataset

4. Results

The developed automated object violation detection and alert system demonstrates highly effective performance across all operational components, validating the integration of YOLOv5 object detection, LBPH face recognition, IoU-based tracking, rule-based violation logic, SMS alert automation, and a real-time Flask web dashboard. During testing, the system successfully captured live video feed from the webcam and processed frames at an average of 10–18 FPS depending on lighting, background clutter, and CPU capabilities, confirming that YOLOv5-based object detection can run smoothly without GPU acceleration. The model consistently detected the target object classes bags, cups, and bottles even when partially occluded, rotated, or located at varying distances from the camera. Precision remained high across different environments, including indoor fluorescent-lit conditions and outdoor natural lighting, confirming robustness.

The bounding boxes produced by YOLOv5 exhibited strong localization accuracy, with minimal false detections. Non-maximum suppression effectively filtered duplicate predictions and contributed to clean detection outputs suitable for downstream processing. The system's face recognition module delivered reliable user identification using the LBPH algorithm, achieving stable recognition accuracy for subjects with adequate face enrollment data. Even under moderate lighting variations, LBPH maintained good confidence scores below the defined threshold, confirming that histogram-based texture extraction is an appropriate and computationally efficient approach for real-time applications. The face-memory buffer further enhanced overall recognition performance by sustaining the last known user identity for a defined duration, ensuring the system correctly linked object violations to the corresponding user even during brief occlusions, profile rotations, or frame drops.

Object tracking using IoU matching demonstrated consistent performance by maintaining track IDs across consecutive frames. The tracking algorithm successfully handled scenarios involving object movement, temporary occlusion, or rapid displacement within the scene. This ability proved essential in distinguishing active violations from non-violations and preventing erroneous SMS alerts. The ground-threshold-based violation logic performed precisely as intended. When an object's centroid crossed below the predefined ground line, the system consistently interpreted this as a littering event, provided the tracked object displayed a downward trajectory. Testing across several trials showed that the violation module successfully identified true violations while avoiding false positives caused by object jitter, camera shake, or momentary bounding-box fluctuations. Once a violation was detected, and the system confirmed an associated user identity from the face-memory buffer, the Twilio SMS alert module activated immediately. The system generated a clear and structured SMS message including the user's name, the type of object detected, and the penalty amount. Message delivery was found to be highly reliable, typically reaching the recipient within 2–4 seconds. This near-instantaneous alerting validates the effectiveness of integrating cloud-based communication services for automated enforcement. The system also logged all relevant user and face information into the SQLite database without errors, confirming database stability across multiple sessions and repeated insertions.

The web-based Flask dashboard provided a smooth and responsive interface for administrators. Live video

streaming via MJPEG performed with minimal latency, allowing operators to visually validate detections and system behavior in real time. User registration, login authentication, and face-training workflows operated without failure. When new users were added, the LBPH model effectively retrained, and the system updated recognition records accordingly. The dashboard's modular design enabled administrators to navigate easily between the video feed, training interface, and system menus. Testing confirmed that the system can operate continuously for extended periods without memory leaks, crashes, or degradation in detection accuracy.

Furthermore, the project successfully demonstrated end-to-end automation: the moment a user is recognized, the object is detected, and the violation is confirmed, the alert is generated—without any need for human intervention at any stage. Such seamless integration shows that the system fulfills all intended objectives of creating a fully autonomous surveillance-based violation detection and penalty mechanism. Overall, the results confirm that the system is robust, accurate, scalable, and suitable for deployment in real-world monitoring environments such as campuses, industries, public corridors, waste-sensitive zones, and restricted areas requiring disciplinary monitoring. The successful performance of each module underlines the practical viability of combining object detection, face recognition, behavioral reasoning, and automated communication into a unified intelligent monitoring solution.

5. Advantages

The proposed system provides a fully automated and real-time solution for monitoring and enforcing cleanliness and safety regulations without human intervention. By integrating YOLOv5 with a lightweight tracking mechanism, the system achieves high detection accuracy with low computational overhead, making it suitable for continuous surveillance. The use of LBPH face recognition ensures reliable identification even under varying lighting conditions and partial occlusions. Automated alert generation through API-based messaging reduces response time and enhances accountability. The system is cost-effective due to the use of lightweight models and an SQLite database, ensuring easy deployment and maintenance. Additionally, the web-based interface enables efficient monitoring, user management, and real-time visualization of events.

6. Future Works

Future enhancements can focus on improving system scalability and accuracy by integrating advanced deep learning-based face recognition models such as CNN-based architectures. The system can be extended to support multi-camera networks for large-scale environments like smart cities. Incorporating edge computing or IoT-based processing can further reduce latency and improve real-time performance. Additional features such as behavior analysis, anomaly detection, and support for a wider range of objects can enhance system capability. Integration with cloud databases and analytics platforms can enable large-scale data storage and predictive insights. Furthermore, implementing multilingual alert systems and mobile application support can improve user accessibility and usability.

7. Conclusion

The development of the automated object violation detection and alert system illustrates the powerful capabilities of integrating modern computer vision techniques, machine learning models, and real-time communication technologies into a unified intelligent surveillance framework. Through the combined use of YOLOv5 for high-accuracy object detection, LBPH for reliable and lightweight face recognition, IoU-based tracking for continuous monitoring of object movement, and rule-based analysis for detecting threshold violations, the system successfully identifies actions such as dropping or throwing bags, cups, and bottles in real time. The addition of instant SMS alerting through cloud-based communication further enhances the system's practical utility by ensuring immediate accountability, reducing human oversight requirements, and reinforcing behavioral discipline among monitored individuals. The Flask-based web interface, paired with an SQLite database, provides an accessible and efficient platform for administrators to manage user registration, face training, and live monitoring. Experimental results confirm that the system performs consistently under varied environmental conditions and achieves stable detection, tracking, and recognition performance without requiring specialized GPU hardware. Overall, the project demonstrates that AI-driven surveillance systems can significantly improve monitoring efficiency, reduce manual intervention, and strengthen rule enforcement in public, institutional, and industrial environments. This work also highlights the potential for future expansion into multi-camera setups, deeper neural face recognition models, enhanced

behavioral analytics, cloud storage integration, and large-scale deployments, making the system an adaptable and scalable solution for next-generation smart security infrastructures.

8. Acknowledgment

We thank you.

The principal, Dr. M. M. Nadakatti, SSIT, Dr. M. C. Chandrashekar, Professor and Head of the ECE Department, Dr. Komala K, Associate Professor and Project Guide in this project. And the project is supported by the Department of Electronics and Communication Engineering at SSIT.

9. References

- [1] L. Ali, H. Baig, A. Khan and M. A. Khan, "Development of YOLOv5-Based Real-Time Smart Vision-Based Lab-Safety Monitoring System," *Sensors*, vol. 22, no. 22, 2022.
- [2] M. Córdova, J. A. Gutierrez, and A. D. P. Moreno, "Litter Detection with Deep Learning: A Comparative Study," *Sensors*, vol. 22, no. 2, 2022.
- [3] N. Zhao, X. Chen and Y. Zhang, "CMCA-YOLO: A Study on a Real-Time Object Detection Model with Context and Multi-Channel Attention," *Electronics*, vol. 13, no. 8, 2024.
- [4] G. Liu, X. Wang and H. Li, "Lightweight object detection algorithm for robots with improved YOLOv5," *Information Processing in Industry (journal)*, 2023.
- [5] Ultralytics, "YOLOv5 — Ultralytics Documentation and Models," Ultralytics docs / GitHub (model documentation and codebase), 2022–2024.
- [6] "What is YOLOv5: A deep look into the internal features of YOLOv5," arXiv preprint, 2024.
- [7] M. Córdova et al., "Litter Detection with Deep Learning: A Comparative Study" (extended dataset & methods), *Sensors*, 2022 — (useful for dataset choices and evaluation metrics).
- [8] J. Bao, "IOU-Guided Siamese Tracking," *Mathematical Problems in Engineering*, 2021.
- [9] K. Yang et al., "IASA: An IoU-aware tracker with adaptive sample assignment," *Journal / Elsevier* (2023) — IoU-aware tracking improvements.
- [10] "Real-Time Object Tracking With YOLOv5 and Recurrent Network on a Hardware Platform," research article (2024) — integration of YOLOv5 + tracking for embedded platforms.

- [11] H. Yang, "Multi-object tracking using Deep SORT and modified pipelines," *Computers and Electronics in Agriculture*, 2022 — applications of DeepSORT and variants for tracking.
- [12] Z. Zou et al., "Deep Learning for Object Detection: A Survey," (survey & review papers, 2021).
- [13] P. Ping, "Smart Street Litter Detection and Classification Based on Deep Learning," *International Journal / World Scientific*, 2020 — litter classification and detection approaches.
- [14] J. Liu, C. Pan and W. Qi Yan, "Litter Detection from Digital Images Using Deep Learning," technical report / conference paper (2022).
- [15] Q. Li et al., "An accurate and adaptable deep learning-based solution for waste detection using Faster R-CNN and attention mechanisms," *Journal / Elsevier*, 2023.
- [16] "A Survey on Face Recognition Systems" (survey — arXiv/2022) — comprehensive review of face recognition techniques (deep and classical).
- [17] A. Budiman et al., "Student attendance with face recognition (LBPH or CNN)," *Procedia / Conference Article*, 2023 — comparative performance of LBPH in attendance systems (practical LBPH evaluation).
- [18] "Human Face Recognition using LBPH" (implementation / paper), 2021 — practical LBPH implementation and evaluation.
- [19] "Development of an Affordable Real-Time IoT-Based Surveillance System with SMS Alerts (Twilio)" — IoT surveillance + Twilio integration (2023).
- [20] D. Nimma et al., "Object detection in real-time video surveillance using improved backbones and transformer heads," *ScienceDirect / 2025* (recent advances combining YOLO backbone + transformer detection heads for surveillance).
- [21] L. W. Batista, S. Khazem, and M. Adibi, "PoTATO: A Dataset for Polarimetric Traces of Afloat Trash Objects," *arXiv preprint*, 2024.
- [22] A. Ahmad, M. U. Rehman, and K. Kim, "Deep Learning-Based Real-Time Waste Classification and Monitoring System," *IEEE Access*, 2023.
- [23] J. Liu, C. Pan, and W.-Q. Yan, "Litter Detection from Digital Images Using Deep Learning," *Int. Journal of Computer Vision & Signal Processing*, 2022.
- [24] R. Harada, T. Oyama, and K. Fujimoto, "Development of an AI-Based Illegal Dumping Trash Detection System," *J. Advanced Mechanical Design, Systems, and Manufacturing*, 2022.
- [25] S. Mahankali, B. Kabbin, and T. A. Kumar, "Identification of Illegal Garbage Dumping with Video," *Proc. IEEE Int. Conf. on Smart Technologies*, 2021.
- [26] M. H. Lee and K. H. Jung, "Detection and Register of Illegal Garbage Dumping Action," *Proc. IEEE Int. Conf. on Consumer Electronics (ICCE)*, 2020