Tech-Sphere Journal of Pure and Applied Sciences (TSJPAS)
A Subsidiary of Tech-Sphere Multidisciplinary International Journal (TSMIJ)
Olufemi et al. Vol 2, Issue 1, 2025 Publication Edition
ISSN: 3092-9598

An Intelligent Image-Based Human Intrusion Detection using ImageAI and Local SMS Alerting

Research Article https://stem.techspherejournal.com

Author Details
Olufemi Ojo[1]*, Abiodun Oguntimilehin.[2], Taiwo Fele [3]
1, 2 Computer Science Department, College of Science, Afe Babalola University Ado-Ekiti, Ekiti State.
3 Computer Science Department, Federal Polytechnic Ado-Ekiti, Ekiti State.

****Corresponding author's email***: jdfemi@gmail.com
DOI: https://doi.org/10.5281/zenodo.16915797

ABSTRACT

Security remains a critical global concern due to the rising rate of crime and intrusions. In response, Video Surveillance Systems (VSS) have evolved from passive monitoring solutions into intelligent, real-time surveillance platforms, powered by advancements in computer vision, machine learning, and telecommunications. Modern systems increasingly aim not only to record footage for forensic review but also to deliver real-time alerts and facilitate immediate intervention. However, a major limitation of many commercial surveillance systems is their inability to accurately classify detected motion. These systems often fail to differentiate between human intruders and non-human movements, leading to inefficient memory usage and frequent false alarms. This paper presents the design and implementation of an Image-Based Intrusion Detection System Using ImageAI and locally triggered SMS notification. The system captures images upon motion detection via surveillance cameras and stores them using the FTP protocol. The stored images are then analyzed using a pre-trained deep learning model to classify and recognize the detected objects. When ImageAI detects a 'person', a flag is set in the MySQL database, which the Diafaan SMS Server polls at intervals to trigger SMS alerts to pre-registered users without requiring internet connectivity. This hybrid architecture provides both cost-effectiveness and timely intervention during live intrusion scenarios while minimizing false positives and unnecessary data storage.

Keywords: VSS, ImageAI, YOLOv3, Object Detection, IDS, SMS.

1 Introduction

Video surveillance systems have undergone a technological shift from traditional, widely used Closed-Circuit Television (CCTV) systems to more advanced solutions such as Drone Surveillance, Robotic Surveillance, Smart Home Surveillance, and Body-Worn Surveillance. CCTV systems, especially, have undergone remarkable evolution, originating from early uses such as the 1940s German missile testing and progressing to widespread deployment in public safety applications from the 1960s onward. What began as simple Closed-Circuit Television (CCTV) systems involved cameras installed at strategic locations to transmit video signals to a central monitoring point. These early implementations lacked recording capabilities and required personnel to continuously monitor video footage. The introduction of reel-to-reel magnetic tape systems marked the beginning of recording functionality, although these were manually intensive, costly, and prone to failure. The emergence of Video Cassette Recorders (VCRs) in the 1970s significantly improved surveillance by enabling rewritable video storage, making systems more practical and reliable.
The 1990s saw another major advancement with the introduction of digital multiplexing, allowing simultaneous recording from multiple video streams and replacing continuous recording with motion-based triggering, thus enhancing

storage efficiency (Huang et al., 2020). Over the past two decades, advances in computing power, networking infrastructure, and artificial intelligence have propelled surveillance from passive recording systems to intelligent, real-time solutions. Modern surveillance systems now incorporate high-definition video, intelligent video analytics, motion detection, facial recognition, remote access, and real-time alerting. These enhancements are designed to address emerging threats and enable rapid response to security breaches (Khan et al., 2021).

Despite these improvements, many traditional systems still rely on post-event forensic review and suffer from high false alarm rates caused by an inability to differentiate between human and non-human objects such as shadows, animals, or swaying trees. This results in excessive storage use and decreased operational efficiency. Researchers argue that an effective surveillance system must not only detect motion but also classify moving objects to distinguish between humans and inanimate entities. Doing so would minimize false alarms and facilitate timely, relevant interventions (Prithviraj & Sengupta, 2016). Ahire et al. (2015) further stress that once human presence is confirmed, the system should immediately trigger alerts to enable rapid action.

The rise of deep learning and computer vision has enabled this next phase of surveillance. Object detection models like You Only Look Once (YOLO), Single Shot Multibox Detector (SSD), and Faster Region-Based Convolutional Neural Networks (Faster R-CNN) now form the core of many smart surveillance systems. The ImageAI library simplifies the integration of these models into Python applications, lowering technical barriers to building intelligent monitoring systems.

This study proposes an intelligent surveillance architecture that addresses false alerts by accurately classifying intrusions using ImageAI. By focusing on identifying human presence within a monitored Region of Interest (ROI) and sending instant SMS alerts through a local gateway, the system avoids the high bandwidth and internet dependency seen in many existing solutions. The objective is to deliver a more practical, cost-effective, privacy- preserving, and ethically responsible intelligent surveillance solution for environments where timely human intervention is critical.

2 Statement of the Problem

In an era of rising security threats, the demand for intelligent and responsive surveillance systems has grown significantly. Traditional video surveillance technologies, while foundational, are increasingly inadequate for real-time threat detection and response. Most existing systems rely primarily on motion detection without the capacity to classify the detected objects. This limitation frequently results in false alarms triggered by irrelevant movements such as shadows, light fluctuations, or environmental disturbances, thereby increasing storage usage, reducing operational efficiency, and desensitizing security personnel to real threats.

Numerous earlier systems, including those by Aminu et al. (2013) and Singla (2014), were built to detect motion through pixel-based differencing and frame thresholding. However, they lacked object classification capabilities, leading to high false-positive rates. Similarly, implementations such as those proposed by Upasana et al. (2015) and Prithviraj and Sengupta (2016) made progress in human detection but still failed to incorporate real-time alert mechanisms that could notify stakeholders during an active intrusion. These shortcomings highlight a persistent gap between detection and timely, actionable response.

Recent developments in deep learning and computer vision have opened new possibilities for smart surveillance. Frameworks like YOLO, R-CNN, and ImageAI enable object classification with impressive accuracy, allowing systems to distinguish between human and non-human objects in real-time. However, many modern implementations require substantial computational resources or rely heavily on continuous internet connectivity for cloud-based alerting, conditions that are not always feasible in low-resource or remote settings.

This research is driven by the need to develop an affordable, efficient, and offline-capable intelligent surveillance solution. By leveraging the ImageAI framework for object classification and integrating a local SMS gateway for real-time alerts, the proposed system ensures that human intrusions are promptly identified and communicated. This approach not only minimizes false alerts but also optimizes memory usage and enables intervention while an intrusion

is actively occurring. The goal is to bridge the technological and operational gaps in existing systems, particularly for environments where cost and connectivity are significant constraints.

3 Research Objectives

3.1 Aim and Objectives

The aim of this study is to design and implement an intelligent image-based human intrusion detection using ImageAI and Local SMS Alerting.

3.2 Objectives of the Study

The specific objectives of the proposed system are to:

a. design a CNN-based system to recognize and classify human objects in images captured from live video sequences using ImageAI.
b. develop a sub-system for human detection alerts using Diafaan SMS Server and Huawei SMS Gateway
c. implement and integrate the systems in (a) and (b).

4 Related Works

Recent advancements in real-time video surveillance have increasingly leveraged deep learning for intelligent object classification and behavioral analysis. Patricia et al. (2025) developed a YOLO-based helmet detection system to address compliance issues in motorcycle parking areas, where manual inspection remains ineffective. Although their approach achieved high real-time detection accuracy, it focused solely on helmet identification, limited dataset and controlled environmental conditions, which may hinder generalization to diverse real-world scenarios. Pallewar, Pawar, and Gaikwad (2023) introduced a CNN-LSTM-based system for detecting human anomalous activities. The major shortcoming of their work is high computational cost and long training time, which limits real-time deployment. This is explicitly acknowledged by the authors, who note that processing the entire video frame instead of focusing only on regions of interest increases latency. Kumar et al. (2025) employed YOLOv8 for personal protective equipment (PPE) detection in industrial environments, emphasizing precise classification for safety monitoring. Khan et al. (2021) implemented a CNN-based multi-object detection framework optimized for lightweight edge devices, enabling efficient real-time processing under constrained computational resources. Huang et al. (2020) applied deep convolutional neural networks to classify human objects in CCTV footage with high precision, particularly in challenging scenarios involving background noise and low lighting conditions. Earlier research in non-deep learning-based systems includes Prithviraj and Sengupta (2016), who combined adaptive background subtraction with object classification to detect human versus non-human motion, reporting 85.7% accuracy. However, their approach did not incorporate real-time notification or proactive alerting. Similarly, Ahire et al. (2015) suggested that systems should not only detect motion but also classify and notify users, though their implementation lacked classification capabilities. Upasana et al. (2015) designed a real-time human motion alert system using webcam input and an Arduino-integrated Bluetooth alert suspension, but their system failed to differentiate between human and non-human entities. Singla (2014) developed a motion detection system using pixel-based frame differencing, which, though effective for static images, resulted in frequent false alerts due to illumination shifts and minor environmental changes.

Aminu et al. (2013) implemented the Motion Detection Security System (MDSS), which used frame thresholding and webcam feeds to detect motion and trigger alerts upon intrusion. Although functional, it lacked object classification and, therefore, generated alerts for irrelevant motion, such as falling leaves or animals such as rats,cats, dogs, etc..

5 Research Methodology

This study adopts a design and implementation-based research methodology. The research work aims to develop a real-time intelligent video surveillance system that reduces false alarms through accurate object classification using the

ImageAI framework. The methodology includes system analysis, system design, software development, integration, and performance evaluation.

5.1 System Design and Architecture

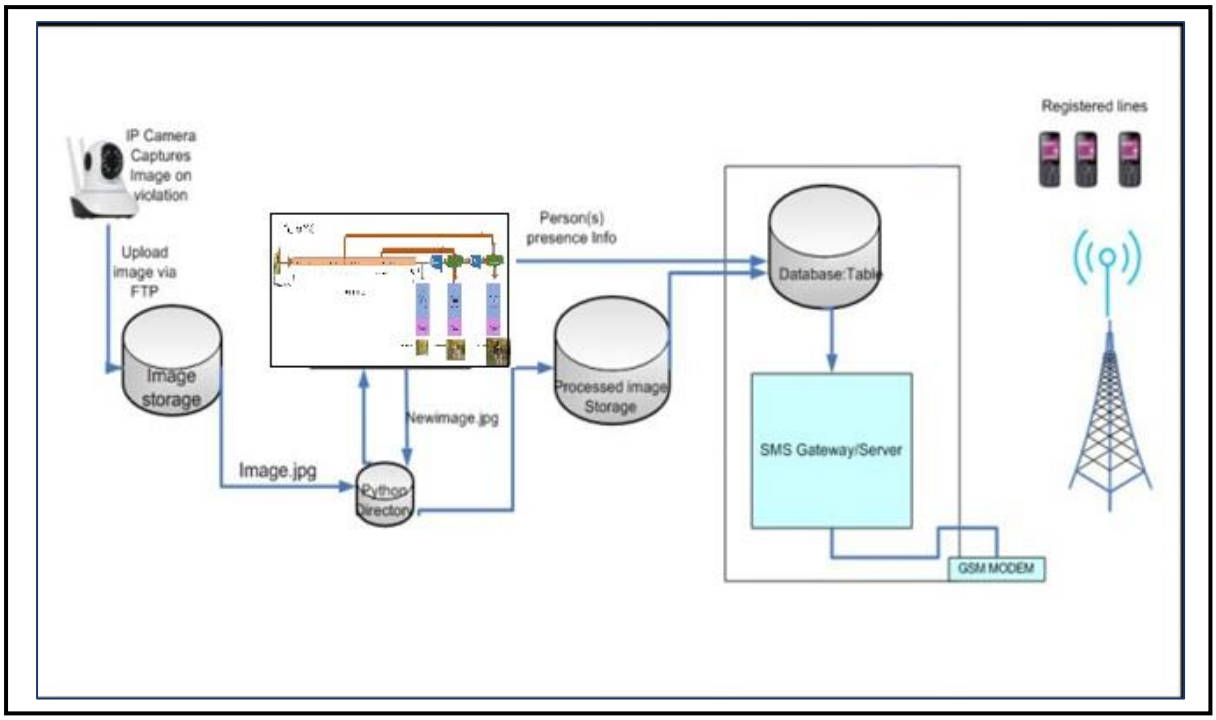

Figure 1: System Architecture

The system integrates both hardware and software components as shown in Figure 1 The main components are outlined below:

i. The hardware architecture consists of a surveillance camera IP-based with NVR, a computer system for real-time video processing, a GSM modem or SMS gateway for alert transmission
ii. The software structure includes Python as the core programming environment, the ImageAI library for object classification, MySQL for event data storage, and an SMS API for delivering intrusion notifications.
iii. The system operates by detecting motion upon instruction, captures image and stored via FTP for further review. Classifying the detected object using ImageAI, and sending an SMS alert if a human object is detected.

5.2 Development Tools and Frameworks

i. ImageAI is used as the primary object detection engine, employing pre-trained deep learning model, YOLOv3 to perform real-time classification.
ii. MySQL serves as the backend database for managing contact lists, logging detection events, and storing relevant metadata.
iii. A GSM modem or SMS gateway facilitates the delivery of alert messages to designated users.

5.2.1 Model Selection and Configuration

The system leverages YOLOv3 through the ImageAI framework due to its balance between detection speed and accuracy. ImageAI supports the use of pre-trained models, which simplifies deployment while ensuring high-performance object recognition (Figure 2).

Tech-Sphere Journal of Pure and Applied Sciences (TSJPAS)
A Subsidiary of Tech-Sphere Multidisciplinary International Journal (TSMIJ)
Olufemi et al. Vol 2, Issue 1, 2025 Publication Edition
ISSN: 3092-9598

5.2.2 YOLO

YOLO is a specialized object detection model that uses CNN layers to (Figure 3):

i. detect multiple objects in a single image
ii. classify them
iii. predict bounding boxes for each detected object

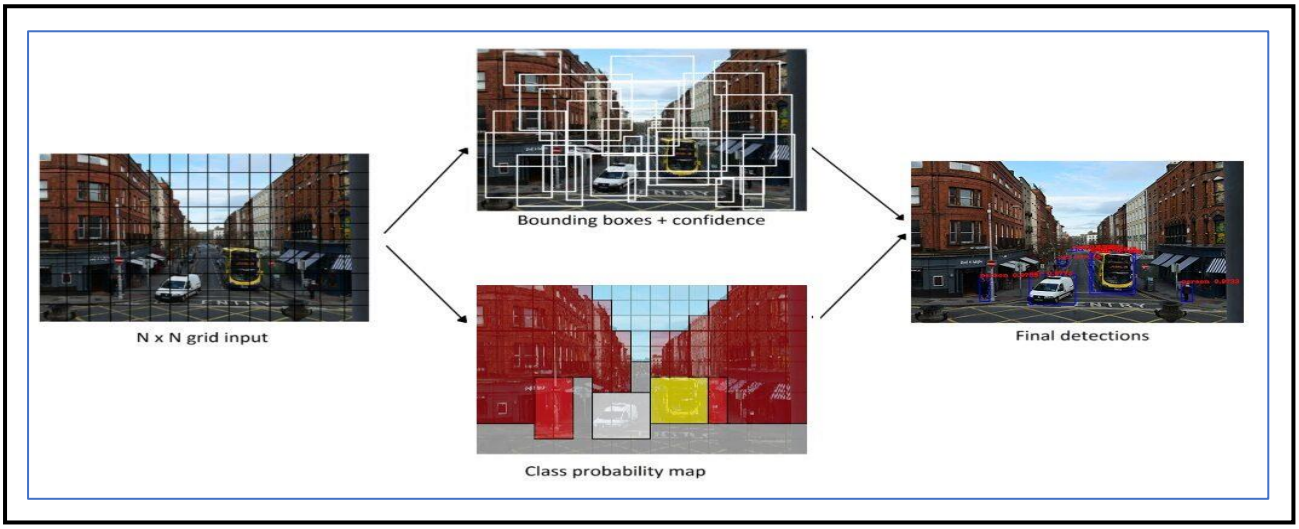

Figure 2: YOLO Operation on an image

Source: https://medium.com/ubiai-nlp/the-evolution-and-applications-of-yolo-object-detection-a-comprehensive-exploration-9b8dbc0ef2a5

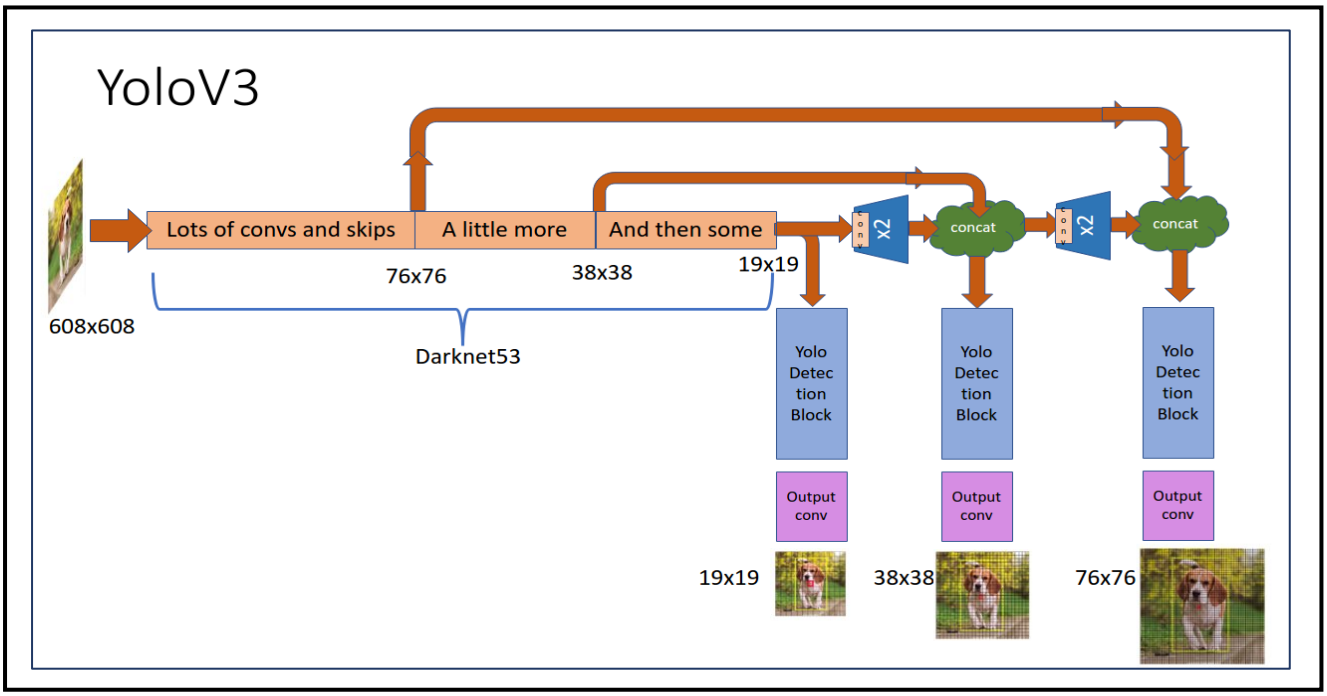

Figure 3: YOLO V3 Sample Framework

Source: https://towardsdatascience.com/yolo-v3-explained-ff5b850390f/

5.3 System Requirement

This applied research is both hardware and software-based. The sub-sections of 5.4.1 will discuss the resources used in this research, detailing the hardware, software tools, and other resources involved in the development of the system.

5.3.1 Hardware Requirements

The hardware resources used in this research are listed below with their respective specifications and features:
The computer system in this design is connected to the Video Network Recorder (NVR) via a switched network. It functions as the storage for image footages and hosts the tools and applications that process video sequences and send SMS alerts.

a. Computer Specifications:

i. CPU: Core i7 - 2.8GHz or more
ii. RAM: 4096 MB
iii. Storage: 500 GB (minimum)
iv. Video Memory: Intel® HD Graphics 4000 – 1792 MB

A Hikvision NVR is used for this research because it captures crisp HD images and video sequences easily. It provides Power over Ethernet (POE) on all Ethernet ports except the one for LAN communications. FTP configuration is also possible over the network.

b. NVR Specifications:

i. Decoding Format: H.265+/H.264/H.264
ii. Channels: 8
iii. Display: HDMI/VGA
iv. Incoming and Outgoing Bandwidth: 256Mbps/160Mbps
v. Recording Resolution: Up to 8 MP

An IP camera was used for the experiment, as POE is available on the NVR, allowing the cameras to be powered by the Ethernet cables that carry the signal.

c. IP Camera Specifications:

i. Colour Type: Colour / Monochrome
ii. Resolution: 2 MP
iii. Motion Activated: Yes
iv. Wide Dynamic Range: Yes
v. Picture Elements (H x V): 1920 x 1080
vi. Image Frame Rate: 25 fps
vii. Network Protocols: TCP/IP, ICMP, HTTP, HTTPS, FTP, DHCP, DNS, DDNS, RTP, RTSP, RTCP, PPPoE, NTP, UPnP, SMTP, SNMP, IGMP, 802.1X, QoS, IPv6, Bonjour
viii. Compression Type: H.264, MJPEG

This device connects to the PC to serve as an SMS gateway, routing triggered SMS messages to the pre-registered numbers. It uses a SIM card from any national GSM provider.

d. Huawei E220 GSM Modem Specifications:

i. Connection: USB 2.0
ii. GSM Frequency Bands: 850, 900, 1800, 1900
iii. UMTS Frequency Bands: 850/1900/2100
iv. Network Protocols: GSM, GPRS, EDGE, UMTS, HSDPA

5.3.2 Software Requirements

a. **Operating System:** Microsoft Windows 10 Professional was used as the operating system and working environment for this research.
b. **Software Development Tool:** The following development tools were installed during the implementation phase:

i. **Python version 3.6 (Minimum System Requirements)**

- Python version: 3.6.x
- Included development tools: conda, conda-env, Jupyter Notebook (IPython)
- Compatible tools: Microsoft Visual Studio, PyCharm
- Included Python packages: NumPy, SciPy, scikit-learn, pandas, Matplotlib, Numba, Intel Threading Building Blocks, pyDAAL, Jupyter, mpi4py, PIP, and others.

Once an image is captured and uploaded to the Python folder, it triggers the execution of the algorithm stored in myfirstdetection.py. This algorithm generates a classified image called new image.

ii. MySQL Server

- If the output from the above processing is TRUE (contains person information), this data is stored in the MySQL database, which the SMS server uses for further processing.

iii. Diafaan SMS Server 4.3

- This module acts as a gateway for sending and receiving SMS. Diafaan SMS Server is used to send and receive SMS using a 3G/4G/GSM modem, SMPP service, analog modem, Visual Basic.NET scripting, and SMTP server. It supports an unlimited number of gateways and connectors. The positive output from the image processing software is connected to the SMS server database.

5.4 System Flowchart

The flowchart for the process design flow is presented in Figure 4.

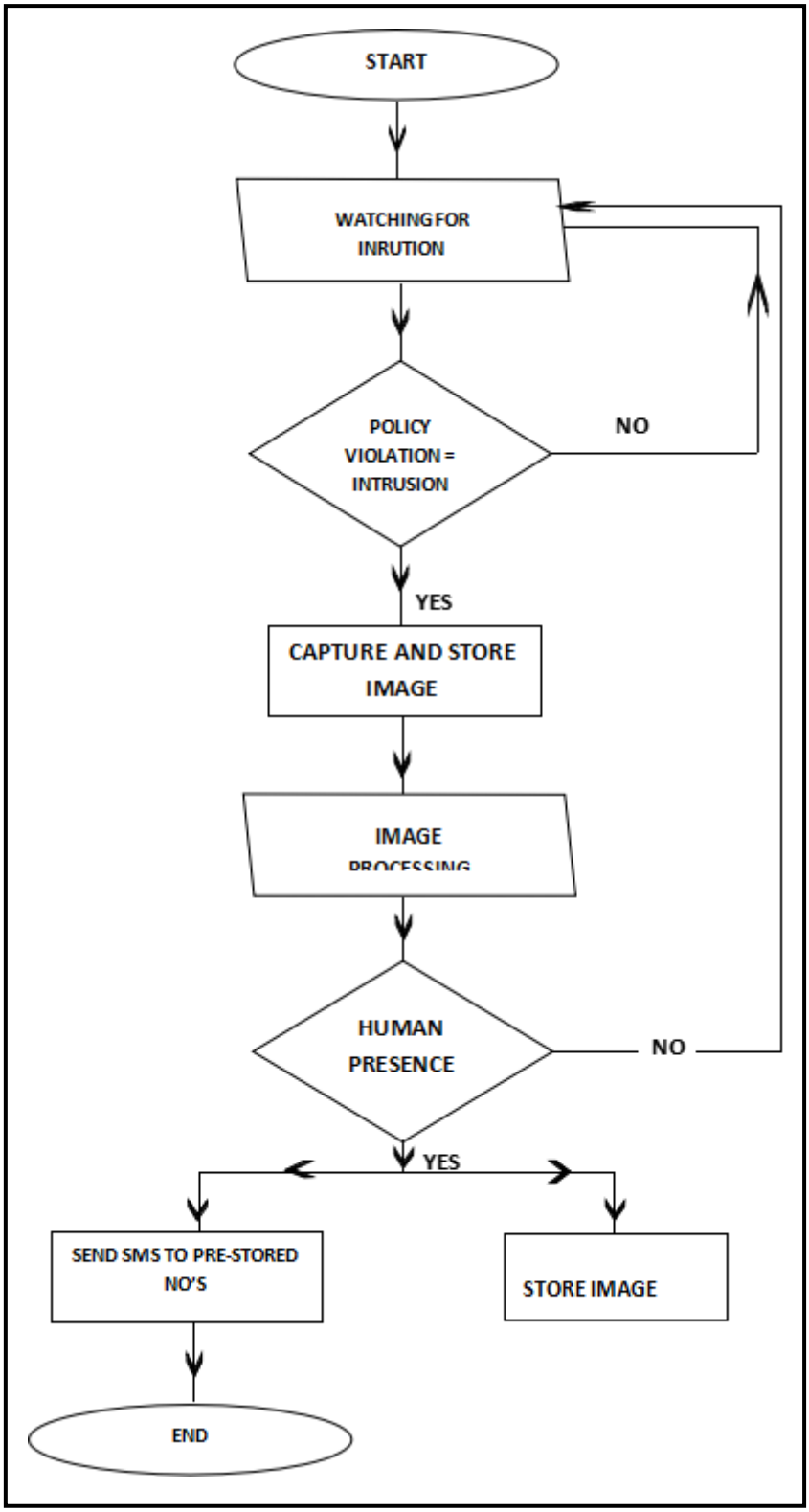

Figure 4: A flowchart showing the process design flow

Tech-Sphere Journal of Pure and Applied Sciences (TSJPAS)
A Subsidiary of Tech-Sphere Multidisciplinary International Journal (TSMIJ)
Olufemi et al. Vol 2, Issue 1, 2025 Publication Edition
ISSN: 3092-9598

6 System Implementation

In this research, a Hikvision IP camera was set up with a POE-enabled NVR DS-7600 Series and a computer on a network segment. The system monitors the facility under surveillance with motion detection recording, particularly during hours when movement is prohibited. A GSM modem, connected to the computer system, acts as an SMS gateway, routing intelligent SMS alerts when motion is detected.

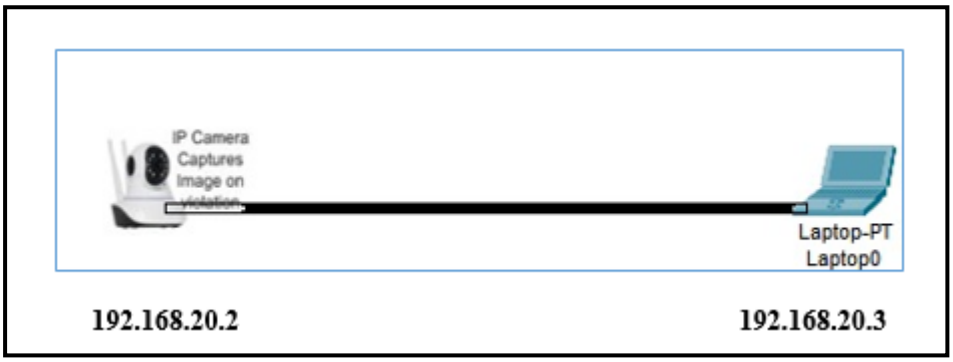

Figure 5: IP Camera communication on IP-based Network

6.1 Installations

The following installations were carried out to create platform for the experiment:
Installation of software tools: - The following software were installed on the PC with specification described above.

a. Install Diafaan SMS Server version 4.3
b. Install MySQL 8.0
c. Python 3.6 was installed on the PC in the python36 folder on the root directory. All dependencies packages such as Tensorflow, numcy, Scipy, Opencv-python, Pillow, Matplotlib, H5py, Keras, and ImageAI were installed from the python environment using the below command one after the other.
    i. "pip install name_of the_pakages"
d. YOLOv3 object classification model was downloaded and place in the python directory in the root directory.

6.2 Hardware Installations

An IP Camera was installed at a required position to monitor a laboratory facility for the purpose of this experiment with a laptop, IP camera and Cat6e cable, as shown in figure 6.1.
The following steps and settings were carried out after the above installations to capture images into a directory on the PC.

1. **Step 1: Login into the Camera from Web Browser**
   The SADP tool, a basic software for finding Hikvision devices on the network, was used to discover the IP address of the camera. The discovered IP address was then entered into the address bar of a web browser (Internet Explorer). After this, a login prompt appeared, and a plugin was installed to enable live video viewing. The browser was closed and reopened with the same IP address. Finally, the camera was logged into using the username "admin" and the password "c@mpsci18".
2. **Step 2: Configure FTP Settings**
   Next, the FTP configuration was set up. This involved entering the server IP, port, username, and password. Additionally, options for uploading pictures, naming the image files, and selecting the directory where captured images would be stored were configured.

3. **Step 3: Motion Detection or Arming**

In this step, the motion detection schedule was set as shown in Figure 5.4 . This configuration ensures that pictures are captured only during motion events within specified time frames. Parameters such as capture interval, image resolution, and image format were also defined. It is important to note that we are not interested in capturing all footage, even though the camera is set to motion detection. We are only interested in motion during certain periods due to the nature of the environment where the experiment was conducted. For instance, the following was set:

- Motion restrictions were applied, specifying areas where no motion should occur during certain times.
- From Monday to Friday, no motion was allowed between 5:00 PM and 6:45 AM.
- From 6:45 AM on Saturday to 6:45 AM on Monday, no motion should be detected.
- If any violation of these rules occurs (motion is detected in restricted periods), the Surveillance System automatically captures images of the objects causing the violation and stores them in the specified directory on the system.

6.3 Image Processing

After the image was uploaded to the folder, it was renamed as image.jpg and moved to the Python directory labeled "python36" in the root directory for processing. The images used for testing, the results, and the output images produced during object classification are shown in Figures 6, 7 and 8, respectively. In this phase, the key object of interest is human. Figure 6 presents a sample image captured by the CCTV camera and uploaded to the Python directory for processing. After being processed with the ImageAI class, Figure 7 shows the console output, which lists the types of objects and their occurrences. This output identifies a backpack and approximately 5 persons out of 6 presents in the image. Finally, Figure 8 shows the resulting image, tagged as New image.jpg in the Python folder. This image contains the original test image with bounding boxes drawn around the recognized objects, along with labeled information.

Figure 6: Sample Testing Image

Tech-Sphere Journal of Pure and Applied Sciences (TSJPAS)
A Subsidiary of Tech-Sphere Multidisciplinary International Journal (TSMIJ)
Olufemi et al. Vol 2, Issue 1, 2025 Publication Edition
ISSN: 3092-9598

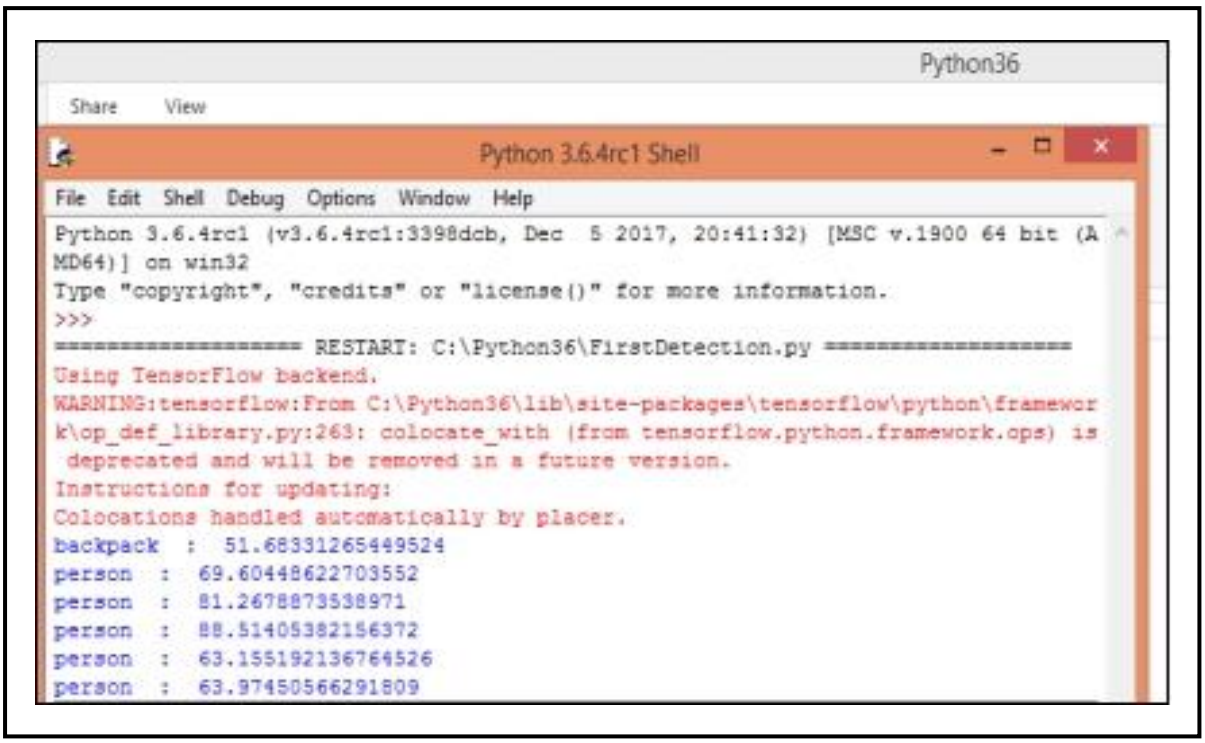

Figure 7: Test-based Result

Figure 8: Processed Image Result

Tech-Sphere Journal of Pure and Applied Sciences (TSJPAS)
A Subsidiary of Tech-Sphere Multidisciplinary International Journal (TSMIJ)
Olufemi et al. Vol 2, Issue 1, 2025 Publication Edition
ISSN: 3092-9598

7 Results and Findings

7.1 Testing with 672 x 392 Resolution images

The first round of testing was carried out on three (3) images with detected intrusions (672 x 392 resolution). These images contained varying degrees of human and non-human objects in different positions (see Figures 9, 10 and 11) ad Figure 12, 13 and 14 shows the corresponding output images. The goal was to test the system's performance, processing time, and the timeliness of SMS delivery to the destination.

a. Input

Figure 9: Sample image input 1

Figure 10: Sample image input 2

Tech-Sphere Journal of Pure and Applied Sciences (TSJPAS)
A Subsidiary of Tech-Sphere Multidisciplinary International Journal (TSMIJ)
Olufemi et al. Vol 2, Issue 1, 2025 Publication Edition
ISSN: 3092-9598

Figure 11: Sample image input 3

b. Output

Figure 12: Output image 1

Tech-Sphere Journal of Pure and Applied Sciences (TSJPAS)
A Subsidiary of Tech-Sphere Multidisciplinary International Journal (TSMIJ)
Olufemi et al. Vol 2, Issue 1, 2025 Publication Edition
ISSN: 3092-9598

Figure 13: Output Image 3

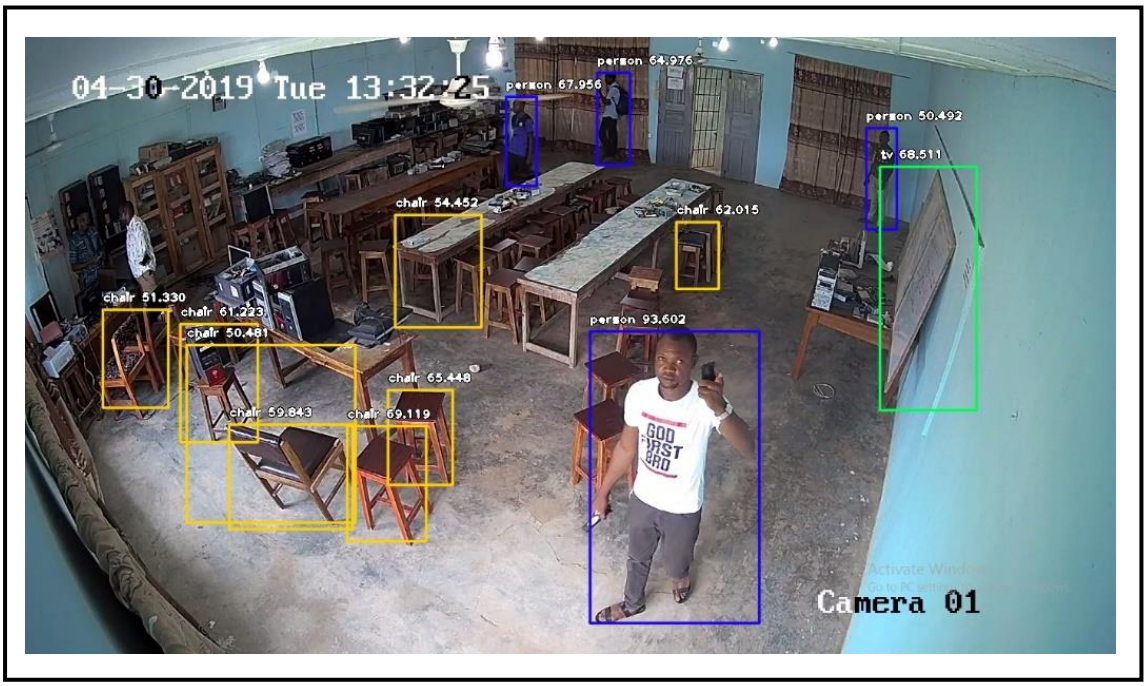

Figure 14: Output Image 3

7.2 Testing with 672 X 392 Resolution images

Table 1: Images processed at 627 x 395 resolution

Image Samples/ resolution	Processing time	Number of objects detected	Number of person(s) detected	Send SMS	Comment
Image1 672x395	37.5 sec	3	0	No	A person was sitting in the image but was not detected.
Image 2 672x395	36.0 Sec	3	2	Yes	100% of persons captured
Image 3 672x395	35.9 Sec	4	0	no	2 people standing were captured.

Table 2: Images processed at 1366 x 768 resolution

Image Samples/ resolution	Processing time	Number of objects detected	Number of person(s) detected	Send SMS	Comment
Image1 1366x768	41.5 sec	13	2	Yes	Persons = 100% captured
Image 2 1366x768	42.9 sec	14	4	Yes	Persons = 67% captured
Image 3 1366x768	43.1 sec	15	6	Yes	Persons = 100% captured

7.3 Observations

Based on the results presented in Tables 1 and 2, the following observations were made:

i. Image Resolution and Object Recognition: Higher image resolutions lead to better object recognition performance by the processing software.
ii. Camera Proximity: Objects positioned closer to the camera are recognized with higher confidence scores than objects that are further away.
iii. Illumination Variance: The quality of illumination significantly impacts object recognition. Better lighting conditions improve the accuracy of the recognition algorithm.
iv. Processing Speed: Images with lower resolution are processed faster compared to images with higher resolution.

7.4 Limitations

This research is limited in the following areas:

i. Relies only on automated image capture.
ii. It uses pre-trained models, which may not be adaptable for all use cases.
iii. The system can only recognize human objects but cannot identify specific individuals.
iv. Testing was not conducted with GPU-optimized hardware.

In future work, these limitations will be addressed by using multiple models for human object recognition and identification, and fine–tuning selected pre-trained models on both public and custom datasets using optimized hardware like GPUs, edge devices, or AI accelerators.

8 Contribution to Knowledge

This research contributes to the field of intelligent video surveillance systems by designing and implementing a real-time object detection system using the ImageAI framework. The key contributions of this work are as follows:

i. Motion Detection and Object Classification: This research creates a system that not only detects movement but also recognizes what objects are present, particularly humans. It helps in identifying specific objects and determining their actions in near real-time.
ii. SMS Alert System: The system sends immediate SMS alerts whenever a human is detected. This is a cost-effective way to notify users immediately about important events.
iii. Using ImageAI for Human Recognition: The study uses ImageAI to recognize human objects quickly and accurately in images, which can improve surveillance efficiency and accuracy.

9 Conclusion

This research makes a significant contribution to the field of intelligent video surveillance systems by designing and implementing an intelligent object detection system using the ImageAI framework (a CNN based architecture). This work presents a robust hybrid architecture for complementing Video Surveillance System, offering both cost-effectiveness and efficient intervention while reducing unnecessary data storage. The integration of ImageAI into the system ensures improved detection capabilities and a reliable alerting mechanism, making it a valuable tool for existing and modern security applications.

References

Ahire, et al. (2015). Real-time security system using human motion detection. International Journal of Computer Science and Mobile Computing, 4(6), 245–250.

Ahmed, M., Ali, S., & Qureshi, R. (2023). Real-time object detection in intelligent surveillance systems using YOLOv5. Journal of Real-Time Image Processing, 20(3), 487–502. https://doi.org/10.1007/s11554-022-01349-4

Amandeep, E., & Goyal, M. (2015). Review: Moving object detection techniques. International Journal of Computer Science and Mobile Computing, 4(8), 345–349.

Aminu et al. (2013). Motion detection security system (MDSS) in live video stream. In International Conference on Artificial Intelligence in Computer Science and ICT (pp. 25–26).

Garg et al. (2020). A review on smart surveillance using deep learning and IoT. Journal of Ambient Intelligence and Humanized Computing, 11(3), 1089–1104. https://doi.org/10.1007/s12652-019-01313-4

Gondchawar, N., & Kawitkar, R. (2022). AI-enabled smart surveillance system for real-time alerting. International Journal of Advanced Research in Computer and Communication Engineering. [Details not provided]

Hapsari, G. C., & Prabuwono, A. S. (2010). Human motion recognition in real-time surveillance systems: A review. Asian Network for Scientific Information, 4(11), 2793–2798.

Huang, J., Wang, M., & Zhang, Y. (2020). Deep convolutional neural network for human object classification in surveillance videos. IEEE Transactions on Industrial Informatics.

Javed, I., Haq, A. U., & Wali, S. (2016, January). Frame differencing flowchart. Retrieved March 2019, from https://www.researchgate.net/figure/Flowchart-of-Frame-Differencing-algorithm_fig5_277328154

Khan, A., Hussain, M., & Shahid, M. (2021). A lightweight CNN-based multi-object detection surveillance system for edge devices. Sensors, 21(7), 2349. https://doi.org/10.3390/s21072349

Kumar, R., Singh, A., & Yadav, V. (2021). Intelligent surveillance system for human detection using deep learning. Materials Today: Proceedings, 47, 2114–2120. https://doi.org/10.1016/j.matpr.2021.06.313

Kumar, S., Sharma, N., & Mehta, V. (2025). YOLOv8-based personal protective equipment detection for industrial safety. Computers & Industrial Engineering.

Li, Z., Zhong, L., & Liu, Y. (2010). Efficient foreground layer extraction in video. College of Computer and Communication Engineering, China University of Petroleum, 11(3), 319–329.

Pallewar, M., Pawar, V. R., & Gaikwad, A. N. (2023). Human Anomalous Activity detection with CNN-LSTM approach . Journal of Integrated Science and Technology, 12(1), 704.

Patricia et al (2025). YOLO-Based Helmet Detection and Intelligent Parking System: A Case Study at Diponegoro University. International Journal of Safety and Security Engineering Vol. 15, No. 5, May, 2025, pp. 1075-1088

Neetha, M. P., & Ariya, T. K. (2014). A survey on object detection, tracking, and identification in video surveillance systems. International Journal of Research in Computer and Communication Technology, 3(1), 82–84.

Olafenwa, M. (2018). Object detection. Retrieved February 2019, from https://towardsdatascience.com/object-detection

OpenCV.com. (2014, November). How to use background subtraction algorithm method. Retrieved from https://docs.opencv.org/3.0beta/doc/tutorials/video/background_subtraction/background_subtraction.html

Patil, S. P. (2016). Techniques and methods for detection and tracking of moving objects in a video. International Journal of Innovative Research in Computer & Communication Engineering, 4(12), 8116–8121.

Safewise. (2018). Motion sensor guide. Retrieved January 18, 2019, from https://www.safewise.com/resources/motion-sensor-guide/

Sharma, A., & Yadav, S. (2022). Human detection in CCTV footage using deep learning approaches. Procedia Computer Science, 199, 676–682. https://doi.org/10.1016/j.procs.2022.01.083

Shrestha, S. (2017). PIR sensors: Are they reliable? Retrieved January 21, 2019, from https://www.tommasociussani.com/blog/2017/09/25/pirsensorsreliable/#disqus_thread

Sikandar, T., & Kamarul, H. G. (2016). A review on human motion detection techniques for ATM-CCTV surveillance systems. International Journal of Computing, Communication & Instrumentation Engineering, 4(6), 213–217.

Singla, N. (2014). Motion detection based on frame differencing. International Journal of Information & Computation Technology, 4(8), 1559–1565