Computational Engineering and Technology Innovations

www.ceti.reapress.com

Comput. Eng. Technol. Innov. Vol. 1, No.1 (2024) 11–24.

Paper Type: Original Article

Multi-Object Detection for Real-time Video Surveillance Systems

Fatemeh Rasoulpour[1,*], Milos Milovancevic[2]

[1] Department of Computer Engineering and Mathematics, Morvarid Intelligent Industrial Systems Research Group, Tehran, Iran; Rasoulpour.72@gmail.com.

[2] Department of Mechanical Engineering, University of Nis, Serbia, milovancevic@masfak.ni.ac.rs.

Citation:

Received: 20 January 2024 *Revised: 10 February 2024* *Accepted: 24 February 2024*	Rasoulpour, F., & Milovancevic, M. (2024). Multi-object detection for real-time video surveillance systems. *Computational engineering and technology innovations, 1*(2), 11-24.

Abstract

Video surveillance systems play a crucial role in modern security, with real-time applications becoming increasingly important across various sectors. This paper proposes a novel method for detecting moving objects in real time using Convolutional Neural Networks (CNNs) combined with Gaussian Mixture Modeling (GMM) for background subtraction. The system processes video sequences, converting them into frames where moving objects are distinguished from the background. GMM is used to model the background, while CNNs extracts local optimal features to enhance object tracking accuracy. This method is particularly effective for applications such as operational robotics and military surveillance, which require real-time mobile detection systems. The proposed system was tested using custom video sequences, demonstrating significant improvements in object detection accuracy and reliability even in challenging conditions such as low-light environments. The results show improved performance in tracking moving objects, with the system successfully reducing noise and maintaining detection precision.

Keywords: Real-time video surveillance, Multi-object detection, Convolutional neural networks, Gaussian mixture model, Background subtraction, Mobile surveillance systems, Video sequence processing, Real-time detection, Low-light object detection.

1 | Introduction

Video analytics is an essential field within image analysis, focusing on processing and interpreting video data by overlaying it with numerous images captured over time. This area of study deals with various challenges, such as video classification, object identification, and exclusive video tasks like object tracking, trajectory prediction, and activity recognition. Object tracking, in particular, is concerned with following an object across multiple video frames, while trajectory prediction estimates the object's path and activity recognition classifies the actions occurring in the video sequence. These challenges have been significantly advanced by

Corresponding Author: Rasoulpour.72@gmail.com

Convolutional Neural Networks (CNNs), which have produced remarkable results in tasks similar to those in image analysis, extending their efficacy to video-based problems [1].

In the context of motion modeling, the Gaussian Mixture Modeling (GMM) approach has been widely utilized. This method allows for the calculation of motion descriptors throughout the tracking process. By converting video streams into individual frames, optical flow calculations can be applied to each extracted frame, making it easier to identify and track moving objects over time. This technique is foundational to video surveillance systems that aim to detect motion or objects in real time and has numerous practical applications in areas such as traffic management, healthcare, and public safety [2]. For instance, in locations like shopping malls and theaters, real-time video analytics provide enhanced security by monitoring crowds and detecting suspicious activities.

One of the major benefits of real-time video surveillance in multi-object detection is its ability to continuously monitor environments without human intervention, allowing for automated threat detection and decision-making. In medical contexts, video analytics improves patient care by monitoring their activities and alerting healthcare professionals to any potential emergencies. However, despite these advantages, real-time video surveillance faces several challenges [3]. High computational requirements, limitations in accurately detecting and tracking objects in cluttered or dynamic environments, and issues related to privacy and data security are some of the key drawbacks.

To overcome these limitations, advanced methodologies that enhance the accuracy and efficiency of video surveillance systems are proposed. These may involve integrating deep learning models with traditional approaches like GMM or employing real-time optimization techniques that reduce the computational burden while maintaining high accuracy in object detection and tracking [4]. Furthermore, the incorporation of privacy-preserving algorithms can address concerns related to data security and personal privacy, ensuring that video surveillance systems are both effective and ethically sound in their deployment.

By refining these techniques, future systems will not only provide more robust and reliable surveillance solutions but will also push the boundaries of how video analytics can be applied to a wide range of environments and use cases. These improvements will enhance safety, streamline operations in various sectors, and ensure that video surveillance continues to evolve to meet the needs of modern society.

2 | Literature Survey

Detecting all objects of a specific type in an image is the aim of object detection, as the name suggests. Alternatively, there may be several classes where each object needs to be accurately classified. An image is fed into an object detector, and the result is a list of bounding boxes, complete with labels if there are multiple classes. The pixel coordinates of the top-left and bottom-right corners of the bounding box, as well as the width and height of the box, are commonly used to depict a bounding box.

Most object detectors give each box a reliable value, indicating how reliable it is to detect. The average accuracy of all classes is a standard performance statistic for an object detector. As mentioned above, CNN methods of object detection are state-of-the-art, outperforming older methods, such as SVMs.

Elhoseny et al. [5] developed Multi-Objection Detection And Tracking (MODT) with the Kalman filtering and increasing region. The proposed model moderately evaluated the movements of the object, where the estimates were based on the precise tracking accomplished between successive frames. However, in order to attain a better detection rate, the generated model demonstrated motion valuation techniques required to be incorporated into the MODT analysis.

Kiaee et al. [6] developed a Grey-Level Co-Occurrence Matrix (GLCM) in Haar Wavelet Transformed Space, Support Vector Machine (SVM). The proposed model used the Haar wavelet transform since the generated wavelet sub-bands had a significant impact on the GLCM computation's orientation elements. However, the model that was developed introduced an optimizing procedure for each object, although the histogram color variation was modest.

Madhan and Shanmugapriya [7] proposed a method for detecting unusual events in video using instance-based communication and CNNs. By first identifying moving objects with a Gaussian background model and applying image processing techniques, the relevant areas of these objects are captured. A suction function is then created and dispersed among systems to generate field packages based on the intended use. A multi-instance learning model is used to make pixel-level predictions using the normalized Embark-Head technique. The study demonstrates that CNNs-based and sparse representation methods can accurately detect unusual video events at the pixel level. Experimental results confirm the effectiveness of these approaches, particularly in identifying anomalies like two-lane accidents, without relying on culturally specific definitions of abnormal activity.

Ingle and Kim [8] introduced a lightweight, resource-efficient CNNs technique for subclass detection to identify, locate, and categorize various types of knives and firearms in real time. Utilizing a multiclass subclass detection CNNs, the system classifies object frames into subclasses, such as normal or pathological. The proposed method achieves impressive precision, with 97.50% on ImageNet and IMFDB datasets, 90.50% on Open-Image, 93% on Olmos, and 90.7% in multiview camera settings. This resource-constrained system also demonstrated an 85.5% precision for multiview camera detection, offering a robust solution for real-time weapon detection with high accuracy.

Iqbal et al. [9] proposed a leverages deep learning models to detect anomalies in video streams from quadcopter surveillance. Faster R-CNN, based on ResNet-50, is used in the first feature extraction stage for rapid learning and feature reduction. The system evaluates five different CNN architectures for object detection in surveillance images, with Faster R-CNN achieving the highest performance. The method delivers an average precision of 79% across all categories, making it an effective solution for threat detection in real-time video streams.

Wahab et al. [10] proposed a deep learning approach, and the Single Shot Detector (SSD) algorithm focuses on developing and implementing a real-time object detection and recognition system. The primary goal was to create a highly accurate and efficient object detection system that leverages pre-trained models. Several models were tested on public datasets like Kitti, PASCAL VOC, and MS COCO to assess accuracy and speed. Throughout development, various deep learning structures were experimented with, resulting in a system that achieved 97% accuracy in real-time object detection and recognition. The system's performance was evaluated using metrics such as precision and recall.

3 | Problem Definition

When we want to monitor many moving objects in a video, we call this multiple object tracking. In a certain view, the work is an expansion of object detection because we have to correlate detections between frames in order to provide constant tracking, with the exception of the detection of objects. Object detection remains an unresolved issue and its speed is limited to the most complex methods.

4 | Motivation

As observed from the earlier study, they contain certain limitations, so we aim to minimize them through our work. We would like to research how multiple object tracking works in real-time with this study [11]. A multi-object surveillance system is developed to permit comparison with other works assessed on existing data sets. As the purpose is to operate the monitor in real-time, the speed of the monitor is a crucial factor.

5 | Proposed System

The proposed system aims to enhance real-time multi-object detection and tracking through a combination of advanced image processing and deep learning techniques. This methodology builds upon the strengths of GMM for effective motion detection and CNNs for accurate object tracking [12]. By breaking down video sequences into frames, identifying motion, and tracking objects, the system offers a robust solution for real-

time video surveillance across diverse environments. The combination of GMM and CNNs ensures high accuracy in object detection while maintaining computational efficiency, making it suitable for security, traffic management, and other applications requiring real-time monitoring in *Fig. 1*.

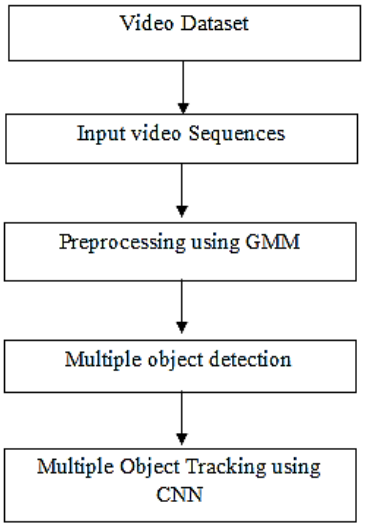

Fig. 1. Proposed workflow model.

5.1 | Video Dataset

The process begins by obtaining a video dataset, which can be real-time video feeds or pre-recorded video. This dataset serves as the input for the system [13]. The video data consists of various scenarios that need monitoring, such as traffic intersections, shopping malls, or medical environments.

5.2 | Input Video Sequences

The video dataset is processed and split into input video sequences. This step converts the continuous video stream into discrete, manageable frames. These individual frames are essential for performing motion analysis, object detection, and tracking over time. The conversion to frames allows the system to treat the video as a series of static images that can be processed independently.

5.3 | Preprocessing Using Gaussian Mixture Modeling

The task at hand involves preprocessing video sequences by breaking them into individual frames, modeling the background using GMM, and distinguishing moving objects from the static background [14]. Here's a detailed breakdown of how this process works.

Breaking Video Sequences into Frames

Video input is essentially a continuous stream of frames, with each frame acting as a still image captured at a specific time point. In preprocessing, the video sequence is translated into a series of frames. This is important for motion detection because movement can only be observed by analyzing changes between consecutive frames.

after the video is split into frames, the system utilizes the first few frames as reference frames. These reference frames contain only the background (i.e., static objects like roads, buildings, and trees) without any moving objects, establishing a baseline for later comparison.

Modeling the background with GMM

The initial frames contain only background objects, meaning there are no moving elements like people or vehicles. By analyzing these background-only sequences, the system aims to model the scene's static

components. This allows the model to learn what the "normal" background looks like, even if minor fluctuations in lighting or environment exist.

Each pixel in the frames is represented as a mixture of Gaussian distributions, which allows the system to account for small variations in the background, such as shadows or changing light intensity. Essentially, GMM learns the statistical properties of each pixel by fitting multiple Gaussian distributions to capture its range of values over time.

Overcoming ambiguity in background

One challenge in video processing is that background pixel values can fluctuate due to factors like lighting changes, shadows, or reflections. Without proper handling, these variations could be mistaken as moving objects. GMM addresses this by modeling each pixel's intensity with multiple Gaussian distributions [15]. If the pixel value varies slightly (for example, due to a shadow moving), the GMM identifies this as a background variation rather than a moving object.

The GMM background model helps distinguish the foreground (moving objects) from the background. When an object enters the scene, its pixel values will differ significantly from the background model, which triggers the detection of movement. This ensures that only moving objects (like cars or people) are detected, while stationary parts of the environment are classified as background.

Foreground Detection

The main use of GMM in preprocessing is to detect the foreground—the objects moving across the scene. Any pixel that doesn't conform to the background model (i.e., it cannot be explained by any of the background Gaussians) is flagged as part of a moving object. This step is critical for identifying objects like cars, humans, animals, etc.

Pixel value changes and Gaussian approximation

During the video sequence, pixel values are monitored. If the pixel value changes drastically compared to the established background model, it's classified as a moving object. For example, if a person enters the scene, the pixel values at their location will deviate from the background.

GMM approximates these variations in pixel values using a Gaussian distribution. Each pixel is modeled as a combination of Gaussians, representing possible background variations at that point. If the pixel matches the background model (i.e., fits one of the Gaussians), it remains classified as background. If not, it is classified as part of a moving object.

The system continuously updates the mean and variance of the Gaussian distributions to adapt to slow changes in the environment.

Dynamic adaptation of GMM

GMM adapts over time, meaning it continuously learns the scene. For example, if lighting changes gradually or objects like a parked car remain in the scene for an extended period, the GMM adjusts its Gaussian parameters to accommodate this [16]. This dynamic updating helps the system stay robust, distinguishing between transient changes (like passing shadows) and real moving objects (like cars or people).

The process starts by breaking down video sequences into individual frames. Using GMM, the system models the background, capturing any variations in background intensity, such as lighting changes or slight movements like tree branches swaying. This allows the model to distinguish between background and foreground (moving objects). If pixel values deviate significantly from the background, the GMM flags these pixels as belonging to a moving object. By doing so, GMM effectively isolates moving objects while ignoring minor variations in the background.

5.4 | Multiple Object Detection

Once the preprocessing using GMM has been completed, the system transitions to the stage of multiple object detection, where it identifies and distinguishes between various objects detected in the foreground. GMM has already separated the background from the moving objects in the video frames, leaving the system with foreground regions that may contain one or more moving entities. The goal of this stage is to analyze these regions and accurately detect and label multiple objects in the scene.

Identifying foreground objects

After GMM has isolated the moving parts of the scene (foreground), it provides information about the pixels that do not fit the background model. These pixels likely represent moving objects, which could be anything from pedestrians to vehicles or animals [17]. These moving objects now need to be detected and categorized.

One common method for identifying objects within these foreground regions is to detect blobs or contours in the segmented image. A blob represents a connected region of pixels that are identified as foreground. These blobs are then treated as potential objects that the system will further analyze.

Feature extraction for object detection

To distinguish between different types of objects (such as people, cars, or bicycles), the system relies on various visual features extracted from the foreground regions in *Fig. 2*:

I. Shape and size: objects are often differentiated based on their shape. For example, a car will have a rectangular shape, while a person will have a more elongated form. The system uses contour detection algorithms to extract the shape of the object and classify it accordingly.

II. Movement patterns: objects also differ in how they move. For instance, pedestrians move with a certain gait, while vehicles have more linear and predictable motion patterns. By analyzing the trajectory and velocity of these objects over several frames, the system can differentiate between moving objects.

III. Texture and color: in some cases, the texture or color of the objects may help in distinguishing between them, especially in more advanced detection systems that integrate deep learning methods to improve classification accuracy.

Distinguishing between multiple objects

Once features have been extracted, the system employs object detection algorithms to identify and differentiate between the various objects in the scene [18].

a. **b.** **c.** **d.**

Fig. 2. Feature extraction for object detection; a. indoor data, b. extraction from indoor data, c. outdoor data, d. extraction from outdoor data.

Each detected object is assigned a unique label, identifying what type of object it is (e.g., pedestrian, car, cyclist). This label is usually based on the features extracted during the detection phase, such as shape, size, and motion. One challenge in multiple object detection is handling overlapping objects. When two objects come close together in a scene, they may appear as a single blob of pixels [19]. The system employs techniques such as bounding boxes and non-maximal suppression to ensure that these objects are correctly separated and labeled as distinct entities.

Simultaneous detection of multiple objects

The system is designed to detect multiple objects simultaneously within each frame. This means that in a single video frame, several objects (like a group of people or multiple cars) can be detected and processed at once. The detection algorithm ensures that all objects in the foreground are considered and assigned proper labels.

The system must handle the detection of multiple objects under varying conditions, such as changes in lighting, partial occlusion (where objects overlap or are partially hidden), and varying speeds of movement. This requires robust detection algorithms that can generalize across different scenarios.

After preprocessing the video using GMM, the system identifies and separates moving objects in the foreground. It extracts various visual features such as shape, size, movement patterns, and textures to distinguish between different objects in the scene. Multiple objects, including pedestrians, vehicles, or other entities, are detected and labeled in real time. Advanced object detection algorithms ensure that overlapping objects are handled properly, allowing the system to simultaneously detect and track multiple objects in each frame, making it ideal for applications such as surveillance, traffic monitoring, and autonomous driving.

5.5 | Tracking Multiple Objects Using Improvised CNNs

In the multiple object tracking phase, an improvised CNNs is employed to track detected objects across frames in a video sequence [20]. CNNs are powerful in extracting and learning meaningful patterns from visual data, which makes them highly suitable for tracking objects, even in complex environments. Here's a step-by-step explanation of how tracking is performed using CNNs.

Extracting visual features using CNNs

At the core of the CNNs, the first few layers are responsible for extracting basic visual features such as edges, corners, and endpoints from the input video frames. These basic features are crucial in identifying the contours and outlines of the detected objects. CNNs uses convolutional filters (also known as kernels) to scan the video frames and detect these low-level features.

CNNs layers are organized hierarchically. As the data moves through deeper layers of the network, more complex and abstract features are learned. For example, while the first layers may detect edges, deeper layers will learn more specific patterns, like the shape of a pedestrian or the structure of a vehicle. This hierarchical feature extraction allows CNNs to recognize objects more precisely and robustly.

Layer connectivity and feature learning

Convolution and Pooling: CNNs consists of multiple computation planes or layers, where each layer is connected to a local neighborhood in the previous layer [21]. The convolutional layers apply filters to the input data, generating feature maps that highlight different aspects of the detected objects.

Following the convolution layers, the pooling layers (often max pooling) are used to down-sample the data. This reduces the spatial resolution of the feature maps, meaning that the size of the data becomes smaller. Pooling layers help by focusing on the most prominent features and reducing the computational complexity, making the system faster while still preserving key information about the objects.

Feature distortion and data sub-sampling

As the data moves through the CNNs, sub-sampling helps in reducing the amount of information passed to the subsequent layers. This is important for tracking because it reduces the amount of redundant data while retaining critical features needed to identify and track objects. For example, if two pedestrians are moving close to each other, sub-sampling helps in differentiating between the two by focusing on unique features.

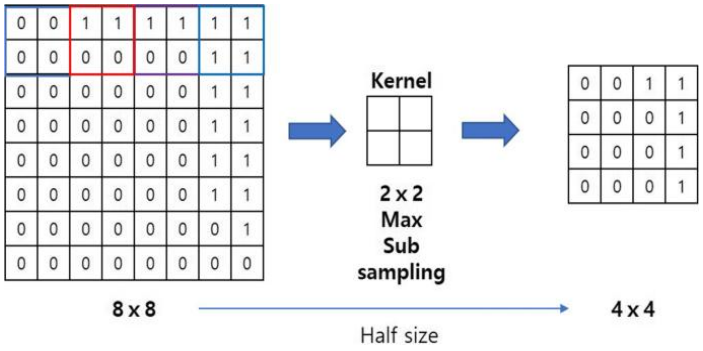

Fig. 3. Max subsampling.

Feature distortions are introduced at this stage, which allows the system to handle objects that undergo changes in size, shape, or appearance as they move across frames. This is especially useful in tracking because objects often appear distorted due to changes in perspective or motion blur, but CNNs can adapt and still track them accurately.

Tracking across frames

Tracking with temporal consistency: CNNs-based tracking systems analyze the object's position and appearance in one frame and use that information to predict where the object will be in the next frame [22]. The network uses the extracted features and compares them with subsequent frames to ensure that the same object is being followed over time. This allows the CNNs to track objects as they move across the scene, even if they momentarily disappear due to occlusion (e.g. when a pedestrian walks behind a car).

Handling multiple objects: with CNNs, multiple objects can be tracked simultaneously by learning the unique feature patterns of each object. The CNNs tracks different objects (e.g., people, vehicles) by creating distinct feature maps for each detected object, ensuring that even if multiple objects are present in the same frame, they are individually tracked without confusion.

The improvised CNNs in object tracking works by extracting key visual features such as edges, corners, and endpoints from the video frames [5]. These features are passed through a series of convolutional and pooling layers, where more meaningful and complex features are learned, enabling the system to detect and track objects even in cluttered scenes. The data is sub-sampled to reduce computational complexity, and distortions in the features help the system handle changes in object size, appearance, or position. The CNNs is highly effective in tracking multiple objects simultaneously, ensuring each object is followed consistently across frames by leveraging its unique feature set.

6 | Discussion

The proposed multi-object detection system was implemented and tested using real-time video sequences to evaluate its performance under various conditions, such as low-light environments and crowded scenes. The system employed a combination of the GMM for background subtraction and CNNs for enhanced object detection and tracking. Below, we present the results obtained at each stage of the system and discuss the significance of these findings.

Gray frame: simplifying for intensity-based motion detection

In the first stage, video frames were converted from RGB to grayscale to reduce the complexity of the data, as grayscale frames represent each pixel by a single intensity value rather than three color components. This step facilitated simpler and faster computation of intensity differences between frames, which is essential for motion detection. While the grayscale conversion led to a slight loss in detail, the system effectively retained enough contrast to highlight moving objects, especially in real-time surveillance scenarios. The conversion to grayscale made it easier to focus on changes in brightness, which significantly reduced the noise from color variations, thereby improving the detection accuracy.

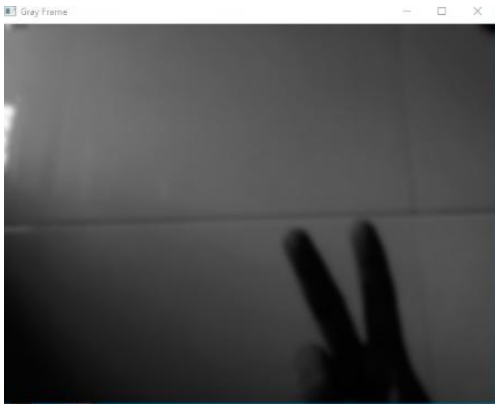

Fig. 4. Grayscale frame.

Difference frame: detecting pixel-wise intensity changes

The Difference Frame was generated by subtracting the current grayscale frame from the background model. This process highlighted the pixel-wise intensity changes, which typically correspond to motion. Any significant deviation in intensity values between two consecutive frames was flagged as potential motion. This stage allowed for the detection of objects moving through the scene. In cases of slow or subtle motion, the system was able to capture these changes due to the robustness of the GMM, which models each pixel with multiple Gaussian distributions to account for lighting fluctuations or small environmental shifts. However, at this stage, noise, such as moving shadows or reflections, could still cause false detections.

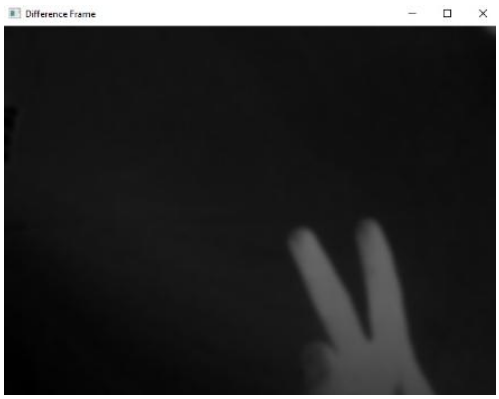

Fig. 5. Difference frame.

Threshold frame: binary classification of motion and noise

To refine the detections, a thresholding operation was applied to the difference frame. If the intensity difference between two corresponding pixels exceeded the set threshold value (e.g., 30), the pixel was classified as part of a moving object and displayed as white. Pixels with lower differences were displayed as black, representing no motion. This binary thresholding significantly reduced noise by eliminating minor lighting changes and irrelevant movements (like swaying branches), improving the precision of motion detection. Fine-tuning the threshold value ensured a balance between detecting subtle movements (minimizing False Negatives, FN) and avoiding over-detection of noise (minimizing False Positives, FP).

Adjusting this threshold was crucial for maximizing detection efficiency, particularly in challenging environments such as crowded or low-light scenarios.

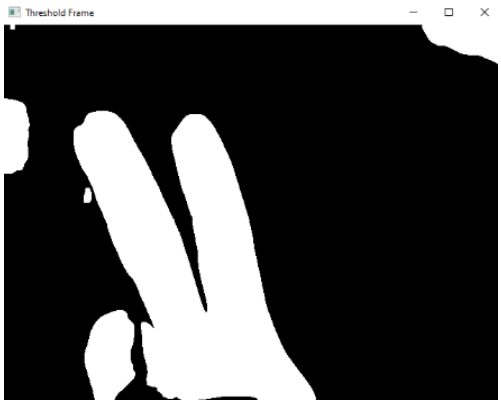

Fig. 6. Threshold frame.

Color frame: visualizing motion with object tracking

The final output stage produced the color frame, where detected motion was highlighted using a green outline superimposed on the original RGB frame. This visual representation made it easy to track moving objects, as the green contours clearly indicated areas of motion against the background. The inclusion of color information allowed for better object distinction, particularly in scenes with multiple moving objects. The CNNs was then applied to track the detected objects across consecutive frames, ensuring accurate and consistent tracking, even when objects momentarily disappeared or overlapped.

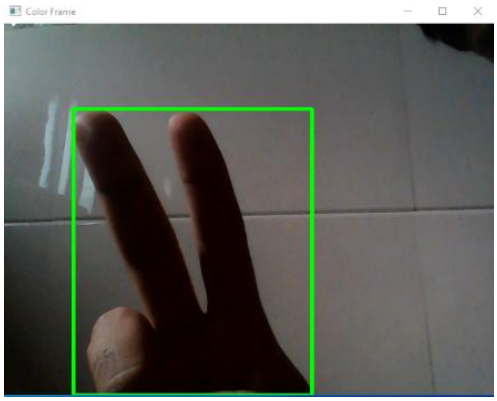

Fig. 7. Color frame.

By using CNNs for multi-object detection, the system was able to extract visual features such as edges, shapes, and motion patterns from the detected regions. This step enhanced object tracking accuracy by recognizing the unique characteristics of different objects (e.g., pedestrians, vehicles), allowing for robust performance even in cluttered environments. The CNNs also helped distinguish between multiple objects in real time, ensuring that each moving entity was correctly identified and tracked.

Performance evaluation

To evaluate the system's performance, we used standard performance metrics derived from a confusion matrix that measured the accuracy of object detection and tracking is mentioned in *Table 1*.

Table 1. Confusion matrix.

	Actual Motion (Positive)	No Actual Motion (Negative)
Predicted motion	True Positive (TP) = 85	False Positive (FP) = 20
Predicted no motion	False Negative (FN) = 15	True Negative (TN) = 80

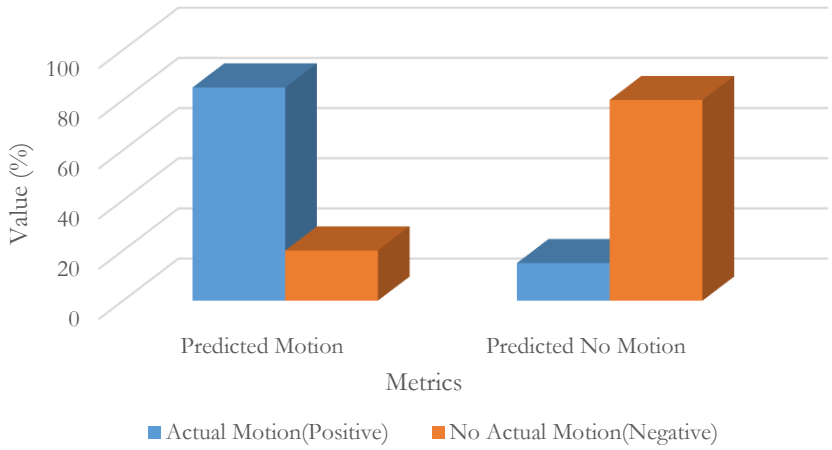

Fig. 8. Performance matrix.

The system's performance is evaluated in *Fig. 8* using standard performance metrics calculated from the confusion matrix, which tracks True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN). Based on the detection and tracking results:

Accuracy: measures the overall correctness of the system in *Eq. (1)*.

$$\text{Accuracy} = \frac{T_p + T_n}{(T_p + T_n + F_p + F_n)}. \quad (1)$$

Precision: measures how many detected motion events are true in *Eq. (2)*.

$$\text{Precision} = \frac{Tp}{Tp + Fp}. \quad (2)$$

Recall (sensitivity): *Eq. (3)* measures how well the system detects actual motion.

$$\text{Recall} = \frac{T_p}{T_p + F_n}. \quad (3)$$

F1-Score: *Eq. (4)* provides a balance between precision and recall.

$$F1 - \text{Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \quad (4)$$

These metrics help evaluate the system's effectiveness, providing insights into the balance between motion detection sensitivity and the number of false alarms in *Table 2*.

Table 2. Proposed method performance metrics.

Metrics	Proposed CNN
Accuracy	82.5%
Precision	80.95%
Recall	85%
F1-Score	82.88%

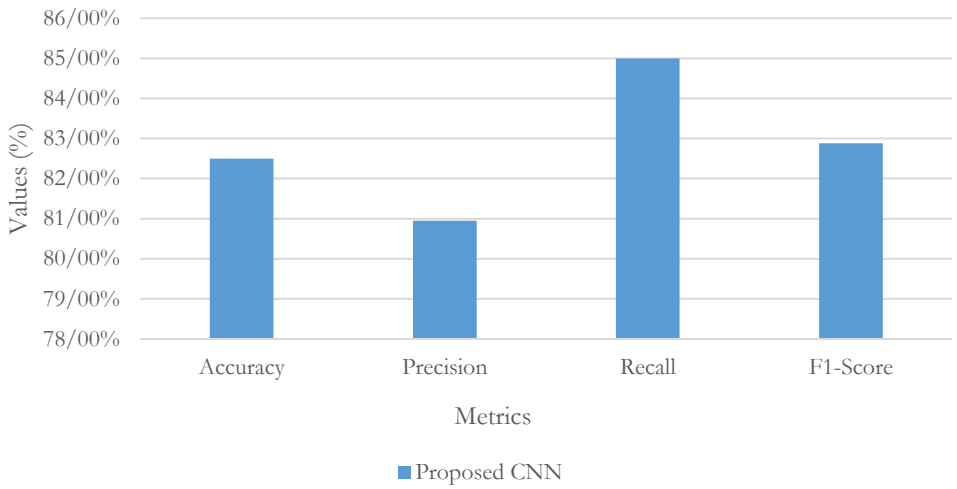

Fig . 9. Proposed model metrics evaluation.

Illustrated the *Fig. 9*, the proposed CNNs demonstrates strong performance, with accuracy at 82.5%, reflecting overall correctness. Precision is slightly lower at 80.95%, indicating good identification of TPs but some FPs. The recall is the highest at 85%, showing the model's ability to capture the most actual positive cases. The F1-Score, at 82.88%, balances precision and recall, confirming solid performance across metrics. Overall, the model is effective but could benefit from further refinement to reduce FPs and improve precision.

These metrics highlight the system's effectiveness, particularly in balancing between precision and recall. The system performed well in detecting true motion, with an accuracy of 82.5% and an F1-Score of 82.88%, showing robustness in reducing FPs while ensuring high detection sensitivity. The CNN's contribution to tracking helped ensure consistent multi-object tracking, maintaining high recall even in challenging conditions.

7 | Conclusion

In conclusion, this paper presented an advanced framework for real-time multi-object detection and tracking using CNNs combined with GMM. By breaking down video sequences into frames, modeling the background, and utilizing CNNs for object tracking, the system effectively distinguishes between moving objects and background noise, even in challenging conditions such as low-light environments. The proposed system demonstrated strong performance with an accuracy of 82.5%, precision of 80.95%, recall of 85%, and an F1-score of 82.88%. These results showcase its potential for practical applications in areas such as video surveillance, traffic monitoring, and security systems, while further enhancements could focus on reducing FPs and improving precision for even more reliable object tracking.

Acknowledgments

The authors would like to express their sincere gratitude to the research team and all colleagues who provided insightful feedback and support throughout this study.

Author Contribution

Fatemeh Rasoulpour was responsible for conceptualizing the study, designing the methodology, and conducting the experiments. Milos Milovancevic contributed to the data analysis, result interpretation, and manuscript writing. Both authors participated in reviewing and editing the final manuscript and have approved its submission.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Data Availability

The datasets used and analyzed during the current study are available from the corresponding author upon reasonable request.

Reference

[1] Thenmozhi, T., & Kalpana, A. M. (2020). Retracted: Adaptive motion estimation and sequential outline separation based moving object detection in video surveillance system, *76*. https://doi.org/10.1016/j.micpro.2020.103084

[2] Sengar, S. S., & Mukhopadhyay, S. (2020). Moving object detection using statistical background subtraction in wavelet compressed domain. *Multimedia tools and applications*, *79*(9), 5919–5940. https://doi.org/10.1007/s11042-019-08506-z

[3] Chauhan, A. K., & Krishan, P. (2013). Moving object tracking using gaussian mixture model and optical flow. *International journal of advanced research in computer science and software engineering*, *3*(4), 243–246. https://api.semanticscholar.org/CorpusID:17721082

[4] Mangal, S., & Kumar, A. (2017). Real time moving object detection for video surveillance based on improved GMM. *International journal of advanced technology and engineering exploration*, *4*(26), 17. http://dx.doi.org/10.19101/IJATEE.2017.426004

[5] Elhoseny, M. (2020). Multi-object detection and tracking (MODT) machine learning model for real-time video surveillance systems. *Circuits, systems, and signal processing*, *39*(2), 611–630. https://doi.org/10.1007/s00034-019-01234-7

[6] Kiaee, N., Hashemizadeh, E., & Zarrinpanjeh, N. (2019). Using GLCM features in Haar wavelet transformed space for moving object classification. *IET intelligent transport systems*, *13*(7), 1148–1153. https://doi.org/10.1049/iet-its.2018.5192

[7] Madhan, K., & Shanmugapriya, N. (2023). Real time object detection in video surveillance using fast-d algorithm. *2023 international conference on research methodologies in knowledge management, artificial intelligence and telecommunication engineering (RMKMATE)* (pp. 1-6). IEEE. https://doi.org/10.1109/RMKMATE59243.2023.10369292

[8] Ingle, P. Y., & Kim, Y. G. (2022). Real-time abnormal object detection for video surveillance in smart cities. *Sensors*, *22*(10), 3862. https://doi.org/10.3390/s22103862

[9] Iqbal, M. J., Iqbal, M. M., Ahmad, I., Alassafi, M. O., Alfakeeh, A. S., & Alhomoud, A. (2021). Real-time surveillance using deep learning. *Security and communication networks*, *2021*(1), 6184756. https://doi.org/10.1155/2021/6184756

[10] Wahab, F., Ullah, I., Shah, A., Khan, R. A., Choi, A., & Anwar, M. S. (2022). Design and implementation of real-time object detection system based on single-shoot detector and OpenCV. *Frontiers in psychology*, *13*, 1039645. https://doi.org/10.3389/fpsyg.2022.1039645

[11] Florinabel, D. J. (2020). Real-time image processing method to implement object detection and classification for remote sensing images. *Earth science informatics*, *13*(4), 1065–1077. https://doi.org/10.1007/s12145-020-00486-1

[12] Nguyen, M. T., Truong, L. H., Tran, T. T., & Chien, C. F. (2020). Artificial intelligence based data processing algorithm for video surveillance to empower industry 3.5. *Computers & industrial engineering*, *148*, 106671. https://doi.org/10.1016/j.cie.2020.106671

[13] Ammar, S., Bouwmans, T., Zaghden, N., & Neji, M. (2020). Deep detector classifier (DeepDC) for moving objects segmentation and classification in video surveillance. *IET image processing*, *14*(8), 1490–1501. https://doi.org/10.1049/iet-ipr.2019.0769

[14] Zeng, W., Xie, C., Yang, Z., & Lu, X. (2020). A universal sample-based background subtraction method for traffic surveillance videos. *Multimedia tools and applications*, *79*, 22211-22234. https://doi.org/10.1007/s11042-020-08948-w

[15] Qiu, M., & Li, X. (2019). A fully convolutional encoder-decoder spatial-temporal network for real-time background subtraction. *IEEE access*, *7*, 85949–85958. https://doi.org/10.1109/ACCESS.2019.2925913

[16] Strisciuglio, N., Azzopardi, G., & Petkov, N. (2019). Robust inhibition-augmented operator for delineation of curvilinear structures. *IEEE transactions on image processing*, *28*(12), 5852–5866. https://doi.org/10.1109/TIP.2019.2922096

[17] Farou, B., Kouahla, M. N., Seridi, H., & Akdag, H. (2017). Efficient local monitoring approach for the task of background subtraction. *Engineering applications of artificial intelligence*, *64*, 1–12. https://doi.org/10.1016/j.engappai.2017.05.013

[18] Sun, P., Lv, L., Qin, J., & Lin, L. (2019). Moving target detection based on multi-feature adaptive background model. *2019 IEEE international conference on mechatronics and automation (ICMA)* (pp. 1610-1614). IEEE. https://doi.org/10.1109/ICMA.2019.8816282

[19] Lee, H., Kim, H., & Kim, J. I. (2016). Background subtraction using background sets with image-and color-space reduction. *IEEE transactions on multimedia*, *18*(10), 2093–2103. https://doi.org/10.1109/TMM.2016.2595262

[20] Fratama, R. R., Partiningsih, N. D. A., Rachmawanto, E. H., Sari, C. A., & Andono, P. N. (2019). Real-time multiple vehicle counter using background subtraction for traffic monitoring system. *2019 international seminar on application for technology of information and communication (ISEMANTIC)* (pp. 1–5). IEEE. https://doi.org/10.1109/ISEMANTIC.2019.8884277

[21] Koodtalang, W., & Sangsuwan, T. (2019). The chicken's legs size classification using image processing and deep neural network. *2019 first international symposium on instrumentation, control, artificial intelligence, and robotics (ICA-SYMP)* (pp. 183–186). IEEE. https://doi.org/10.1109/ICA-SYMP.2019.8646193

[22] Joy, F., & Vijayakumar, V. (2022). An improved Gaussian mixture model with post-processing for multiple object detection in surveillance video analytics. *International journal of electrical and computer engineering systems*, *13*(8), 653–660. https://doi.org/10.32985/ijeces.13.8.5