A Novel Optimization Algorithm for Smart Video Surveillance System and Change Object Detection

Fahad Siddiqui[1(✉)] and Shafaq Siddiqi[2]

[1] Sukkur IBA University, Sukkur, Pakistan
fahad.siddiqui@iba-suk.edu.pk
[2] Graz University of Technology, Graz, Austria
shafaq.siddiqi@tugraz.at

Abstract. Security has been a significant concern in every aspect of life. Traditional surveillance systems require a huge amount of storage capacity to save recorded videos. Moreover, as these systems are not fully automated, object detection and change detection need a lot of computational power in real-time surveillance. Finding a specific event or object in the recorded videos becomes more tedious and time-consuming. Since the last decade, researchers and developers have been working on the optimization and improvement of surveillance systems. This paper mainly focuses on surveillance system optimization by presenting an algorithm that not only optimizes objects and change detection but also requires comparatively less time in searching for a particular object from the library of recorded videos. The paper also presents a software application that offers various unique features such as searching for a specific object and on-time notification on targeted object detection.

Keywords: Object detection · surveillance system · video security · YOLOv5

1 Introduction

Nowadays, surveillance systems are getting more and more popular because the government, public and private organizations are using them to keep a check on various aspects of safety and security [4,24,29,33,38]. With the advancement in technology, the whole concept of video has changed [19] and reached a dimension of modern digital output that not only provides high-quality videos but also enhances interactive features. The impact of high-quality video resulted in high storage space as video recording in HD (720px) at the rate of 30fps normally takes up to 86 GB per day ([1,26]), and normally surveillance systems store recording for months. So, the need for storage to record and keep those high-resolution videos has increased, raising the cost of buying several storage devices.

Most of the surveillance systems today are incapable of making decisions in real-time [28] and unable to decide when to record, what to detect, and what

T. Guarda et al. (Eds.): ARTIIS 2023, CCIS 1935, pp. 103–117, 2024.
https://doi.org/10.1007/978-3-031-48858-0_9

to ignore. These systems require humans to continuously monitor screens for security reasons [10]. Moreover, advancement in surveillance systems and an increased number of cameras resulted in high labor costs, useless recorded video frames, no track of change, and limitation of attention for multi-screen monitoring [32]. However, improvements in computing power, availability of large-capacity storage devices, and high-speed network infrastructure paved the way towards more robust smart video surveillance systems for security [10,25].

Traditional surveillance systems are passive and thus keep recording continuously, increasing the cost of storage. Also, a particular object/event detection in these systems is a computationally costly and tedious job as it needs to go through the whole video track to analyze high-resolution video images [41]. However, there are few systems that perform object detection intelligently using deep learning [14,16,20,37,42] but they need high computation power for continuous processing on streaming. In contrast, machine learning-based systems could be less computationally intense but training with high accuracy is challenging in these systems as it requires huge labeled data and manual feature extraction [8].

Some systems continuously classify each captured frame before saving it into a video, increasing the number of objects in an image. It thus results in a surge of processing time [2] and requires a lot of computational resources. Moreover, the issue with these approaches is that even after classifying every image, they are unable to detect changed objects in the camera frame. This is not their incapability, but these systems are not designed to detect any change in objects which can probably miss the optimization. Additionally, due to a lack of real-time object detection, these systems do not send on-time notifications when a particular type of object is targeted. Therefore, there is a need for an efficient algorithm, which can perform real-time object detection in an optimal manner, and a system that requires comparatively less storage as well as computational power and provides intelligent object searching capability and unattended surveillance by sending an on-time notification when a particular object is detected, or a new object entered the scene.

This paper presents an algorithm that not only optimizes objects and change detection but also requires comparatively less time and effort in searching for a particular object from the library of recorded videos. The paper also presents an automated surveillance system, namely Smart Video Surveillance System (SVS System), which provides a solution for real-time surveillance.

We present the state-of-the-art in Sect. 2, and the architecture of our surveillance system and optimizations in Sect. 3 and Sect. 4 respectively. In the end, we discuss our experiments in Sect. 5.

2 Literature Review

Zhiqing Zhou et al. [43] presented optimizations in wireless video surveillance systems. The main idea was to create a modular system that reduces network complexity and optimizes overall performance. Mohammad Alsmirat et al. [3] proposed a framework that performs optimization for efficiently utilizing resources

of automated surveillance systems on the edge server. For wireless bandwidth optimization, Proportional Integral Differential (PID) technique is used.

Wang and Zhao [39] and others [11,13] proposed a motion detection technique that is based on background subtraction. In this technique, a series of video images had been taken, and these images contained geometrical information of any target. Thus, relevant information is extracted for analysis and motion detection. This technique greatly improved the compression ratio.

Devi et al. [36] presented a motion detection algorithm based on background frame matching. This was a much more efficient method for motion detection. It required two frames one was a reference frame another was an input frame. Moreover, the reference frame opted to compare with the input frame and their difference in pixel values determined motion.

Nishu Singla [35] presented another technique for motion detection which used consecutive frame differencing. A reference frame was used for differencing with the current/input frame and pixel-based difference produced holes in the motion area. After that, a transformation (RGB to Gray) was applied to highlight the motion area and then another transformation (Binarizing) was applied for highlighting the motion area. The limitation of this approach was determining air effect as motion which is absolutely not acceptable for surveillance systems.

Chandana S [6] presented two more techniques for motion detection and stored video based on motion detection. The first technique was using normalized cross-correlation to find the similarity between two frames. The second technique was to calculate the sum of absolute differences between two consecutive frames.

Zhuang Miao et al. [23] presented an intelligent video surveillance system that worked on moving object detection and tracking. For object detection, three consecutive frame differencing technique was used whereas mean shift was used for tracking. Similarly, Zhengya Xu and Hong Ren Wu [21] proposed a real-time video surveillance system based on multi-camera view and moving object tracking. Moreover, other functionalities of the camera like zoom/pan/tilt for static cameras kept intact and static background modeling was used to analyze and track objects.

A. A. Shafie et al. [34] and others [7,9] presented a video surveillance system for traffic control. Basically, different vehicles were detected in real-time using blob segmentation. Every time, a new vehicle came in the range of the camera, blob segmentation drew a boundary box after classifying it.

K. Kalirajan and M. Sudha [17] proposed an object detection-based surveillance system for detecting moving objects and then performed localization for classification. For object classification, the system is used to separate background and foreground and perform classification for foreground using the Bayesian rule.

Kyungnam Kim and Larry S. Davis [18] presented another object detection methodology for real-time object detection and tracking. For object detection, background subtraction was used, and tracking was performed. Moreover, multi-camera segmentation was also implemented for parallel classification. Anima Pramanik et al. [27] presented an approach for stream processing. Frames were

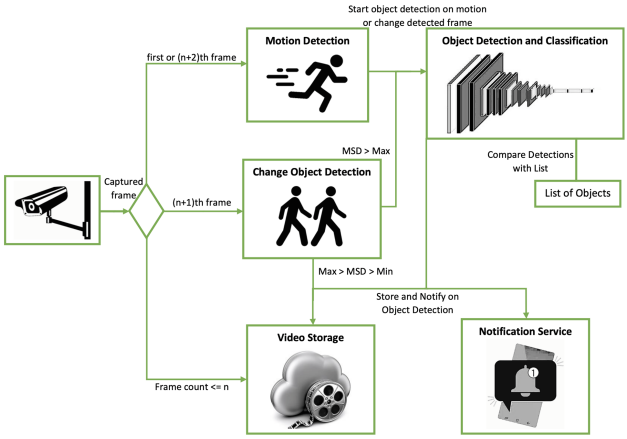

Fig. 1. Overview of SVS System.

extracted from the stream and analyzed for object detection after that they were passed for feature extraction and anomaly detection.

Hanbin Luoa et al. [22] presented a surveillance system that can detect hazards and dangerous areas in construction sites. YOLOv2 was used to detect objects and predict boundaries and proximity was calculated between people and detected objects other than humans.

Hyochang Ahn1 & Han-Jin Cho1 [2] identified that Convolutional Neural Network (CNN) based real-time object detection models are very computationally intense and face difficulty in processing every frame. Also, presented another approach with background subtraction using machine learning for real-time object detection.

Motivated by the extensive state-of-the-art, our proposed solution also builds upon the proposed technique with novel optimizations which results in less storage and runtime.

3 Design and Architecture

The Smart Video Surveillance System has Component-Based architecture and has different functionalities which are covered in distinct and independent components. The system is divided into five main components namely motion detection, change object detection, object detection and classification, video storage, and notification service.

The SVS System connects through a wired or wireless connection to the camera. Figure 1 The system captures the frame and senses motion with the motion detection component and if it finds any motion, it activates the object detection and classification component. The object detection and classification component classifies and compares the predicted output to the provided list of targeted objects. In case of the detected object matches with the targeted

objects, this component will send a request to the Video Storage component to store the frames and Notification Service to trigger notification for the user. Once the recording is started, the Video Storage component will store n number of frames then pass control to change the object detection component. The change object detection will sense if there is any change in the object and based on this the control will be passed to either the Video Storage component to store n frames again or the object detection and classification component to detect new objects.

4 Methodology

In this section, we discuss the implementation details of each component with proposed optimizations and choices of parameters.

4.1 Motion Detection

Moving object detection is widely performed by taking the pixel-wise difference between the input frame and reference frame. Many improvements have been proposed to existing frame differencing and background subtraction approaches. However, these approaches have several limitations e.g., air effect, illumination effect, which may lead to unwanted results.

The proposed approach for motion detection is calculating Mean Squared Deviation (MSD). For two consecutive frames, we apply a threshold on MSD to decide a change in frames is motion or not. Moreover, to overcome the limitations of previous approaches, frames are converted into grayscale before calculating MSD which eliminates illumination and color effects (see Algorithm 1). The threshold value is set after an experimental evaluation. The reason for choosing MSD for motion detection is its fast execution and avalanche effect in minor changes. Moreover, MSD equal to zero implies similar frames, which means no motion is detected whereas increasing MSD from zero defines the intensity of dissimilarity between two consecutive frames. If this dissimilarity surpasses the threshold, it will be interpreted as motion is detected. The mathematical representation for calculating MSD is following:

$$MSD = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I_{\text{current}}(i,j) - I_{\text{previous}}(i,j)]^2$$

This algorithm first takes two grayscale frames, calculate their MSD based on the approach mentioned above, and compares the obtained MSD with a preset threshold to make any decision on motion detection.

4.2 Change Object Detection

The Change Object Detection component is one of the key components of this system. Motion Detection determines the presence of any moving object in front

Algorithm 1. Motion Detection in Frames

Input: *Current Frame curr, Previous frame prev*
Output: *Boolean*

1: $curr \leftarrow$ convert current frame into grayscale
2: $prev \leftarrow$ convert previous frame into grayscale
3: $m, n \leftarrow curr.shape()$ *//returns current image shape*
4: $MSD \leftarrow \frac{\sum(curr \quad - \quad prev)}{m \times n}$
5: **if** $MSD > Threshold$ **then**
6: **return** True *// motion detected*
7: **else**
8: **return** False *// no motion detected*
9: **end if**

of the camera. But, once a cycle of recording has been completed, the system will again start the new cycle from motion detection to video recording which may cause a high computational cost. To further optimize this process, Min-Max thresholding has been introduced through that the change of object will be determined and in case of the same object(s) is present in the range of the camera, the system will start recording again by eliminating the computationally expensive classification task.

Min-Max thresholding is the criteria for the Change Object Detection component to decide whether a new object has been detected in the range of the camera or the existing object(s) is still present there. In this thresholding, Min is the minimum threshold that is necessary to qualify to call any changes in the frames as 'motion'. In contrast, Max is the minimum threshold on which any change in the frames will be considered either the same object presence or a change of object(s) in the current frame.

Moreover, the Change Object Detection will apply Min-Max thresholding to determine the cause of change and decide the decision as follows:

$$MAX > MSD > MIN \quad \textit{Same object detection}$$
$$MAX > MSD \quad \textit{Changed object detection}$$

Change Object Detection details are given in Algorithm 2. This algorithm takes two grayscale frames and calculates its MSD similar to Algorithm 1. After obtaining MSD, it applies Min-Max thresholding to conclude if the same object is present, object has changed or there is no motion at all.

4.3 Object Detection and Classification

Smart Video Surveillance System (SVS System) uses a motion detection approach before starting real-time object classification - classifying an image with a multi-class classifier during live streaming. Once the motion detection component responds to the change in frames as motion, the next task will be to find

Algorithm 2. Change Object Detection in Frames

Input: *Current Frame curr, Previous frame prev*
Output: *Boolean(min, max)*

1: *curr* ← convert current frame into grayscale
2: *prev* ← convert previous frame into grayscale
3: *m*, *n* ← *curr.shape() //returns current image shape*
4: $MSD \leftarrow \frac{\sum(curr \quad - \quad prev)}{m \times n}$
5: **if** $Max > MSD > Min$) **then**
6: **return** (True, True) *//same object*
7: **else if** $MSD > Max$) **then**
8: **return** (True, False) *//changed object*
9: **else**
10: **return** (False, False) *//No motion detected*
11: **end if**

object(s) that has caused the motion. In this regard, a multi-class classifier is needed which can process frames and return the detections efficiently.

Object detection and classification are valuable but computationally expensive in surveillance systems. There are many approaches are mentioned in the literature review for real-time object detection and a detailed comparison in terms of time and accuracy of different models including YOLO (You Only Look Once), SSD (Single Shot Detection), and R-FCN (Region-based Fully Convolutional Networks) is presented in [40]. However, they all need powerful machines with Graphical Processing Units (GPU) for surveillance systems. The proposed approach for object detection and classification is to use the You Only Look Once version 5 (YOLOv5) algorithm [15,30] based on trade-off [12,40] that is balanced in terms of speed and accuracy in real-time processing. Algorithm 3 presents our object detection and classification algorithm that loads the YOLOv5 model and set its type. Then it starts detection with the provided parameters.

Algorithm 3. Change Object Detection in Frames

Input: *image*
Output: *imageWithDetections*

1: *detector* ← *ObjectDetection()*
2: *detector.setModelTypeAsYOLOv5()*
3: *detector.setModelPath("my_YOLO_model.h5") //loading weights*
4: *detections* ← *detector.detectObjectsFromImage(params)*
5: **return** detections

4.4 Video Storage

While YOLOv5 is fast and accurate in real-time classification, it requires GPU to process every frame continuously. As a result, processors keep busy in classification on streaming which makes surveillance systems computational-intense.

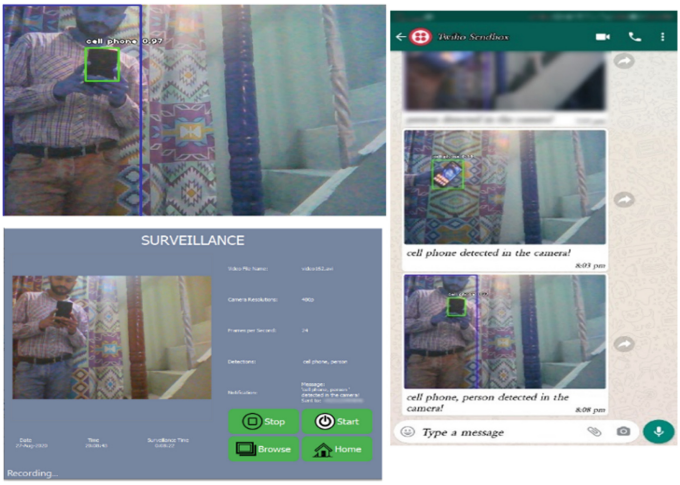

Fig. 2. Interface of SVS System.

Hence, there is a need for optimization so that surveillance systems with real-time object detection can be prevented from high computation and may also be run on computers without GPUs. The proposed approach for this optimization is to classify a single frame (if and only if motion is detected) and continuously store n number of frames if any of the specified objects are found in the classified frame. In other words, the system does not classify every frame during streaming (until there is motion and detections does not contain the specified object(s)) instead it classifies a single frame after recording n number of frames. To prevent security risks, $n-1$ frames are stored directly until the next classification is performed, if classification activates on motion detection. This approach has two limitations. First, the value of 'n' is experimental and may vary from application to application. This system is designed for home premises, but different applications may have different values of 'n'. The experiment details are shown in the experiments and threshold value section. Second, if a system detects motion continuously but none of the user's specified objects is detected in the frame, the system will start to classify every frame just like many surveillance systems.

4.5 Notification Service

SVS System has additional on-time notification services functionality which is not yet implemented in existing surveillance systems. This system allows the user to select notify option for any object and the system would generate a notification for the user if an object marked notify is detected and sends it to the user's cell phone via SMS (short message service) and send a classified image to the user's WhatsApp number. Fig 2 shows the interface and notification sample of our surveillance system.

Table 1. MSD score on different walking motions

Direction	Speed	MSD
Left to Right & Right to Left	Slow Walk	89
Left to Right & Right to Left	Normal Walk	127
Left to Right & Right to Left	Running	223
Front to Back & Back to Front	Slow Walk	66
Front to Back & Back to Front	Normal Walk	80
Front to Back & Back to Front	Running	114
Diagonally	Slow Walk	170
Diagonally	Normal Walk	287
Diagonally	Running	366

5 Experiments

This system is developed in Python 3.7 using OpenCV [5] library. The experiments are conducted on non-GPU Intel®Core™ i5-3230M CPU 2.60GHz (4CPUs) 3rd generation with 8192MB RAM and 300 GB hard disk drive.

5.1 Motion Detection Threshold

To determine the threshold for motion detection, we conducted a series of experiments. These experiments include a person's movement across the camera, throwing an object in the range of the camera, different light conditions (low, medium, high, daylight), Movement of hung clothes due to air effect and raining or water flow in front of the camera.

Experiment 1 Person Movement: In this experiment, person movement is observed. A person moved from left to right, right to left, front to back, back to front and diagonally with different speeds such as slow walking, normal walking and running. We observed that walking diagonally in front of the camera always results in higher deviation no matter the pace of walking (see Table 1).

Experiment 2 Light Effect: An observation is taken in different light conditions in front of the camera. The light conditions were low (toward darkness), medium (normal/day light on a clear sky with a temperature of 22°C) and high (very bright light a standard flashlight in a studio). Table 2 shows that MSD is very high when transitioning from high light to low and vice versa. In contrast, the MSD is very low when this transition happens in medium to low light.

Experiment 3 Throwing Objects: In this experiment, two different objects of different masses are thrown in front of the camera. The objects were an ordinary

Table 2. MSD on different light conditions

Light Condition	MSD
Low to Medium	80
Low to High	213
Medium to Low	60
Medium to High	75
High to Low	229
High to Medium	70

Table 3. MSD on different objects

Direction	Speed	MSD
Pen	Slow	30
Pen	Normal	40
Pen	Fast	60
Box	Slow	34
Box	Normal	50
Box	Fast	83

pen and a box of 10 cm * 10 cm. Table 3 show a deviation of 60 and 83 for both pen and box respectively. We repeated the experiment three times and observed higher deviation when an object of a bigger mass is thrown toward the camera.

Experiment 4 Air Effect: In the end, the air effect over the curtain and tree is evaluated in indoor and outdoor scenes. The results (Table 4) show a directly proportional relation between high air pressure and MSD. The more air pressure increases, the more MSD value increases. Similarly, the lower air pressure will result in a lower MSD value.

Table 4. MSD on different air conditions

Scene	Air Intensity	MSD
Indoor	Low	50
Indoor	High	65
Outdoor	Low	60
Outdoor	High	80

We also performed an experiment with dropping water in front of the camera and on a rainy day but did not find a significant difference in MSD on a light rainy

day. On a heavy rainy day, the recorded MSD was 52. We use this information to increase the threshold value if a day is predicted as rainy. After getting the MSD for different scenarios via our experiments we took the average of these MSD values on points when a motion was detected and when a motion was not detected. Using these average values, we found the min and max values for Min-Max thresholding as **90** and **270** respectively.

Table 5. Number of stored frames between classifications.

Camera	Direction	Speed	# Frames	Time (s)
Camera 1	Left to Right	Slow	175	41.10
Camera 1	Front to back	Slow	135	30.24
Camera 1	Diagonally	Slow	140	31.20
Camera 1	Left to Right	Medium	110	25.38
Camera 1	Front to back	Medium	85	20.58
Camera 1	Diagonally	Medium	100	23.34
Camera 1	Left to Right	Fast	70	17.70
Camera 1	Front to back	Fast	70	17.40
Camera 1	Diagonally	Fast	75	18.60
Camera 2	Left to Right	Slow	600	30.00
Camera 2	Front to back	Slow	480	24.00
Camera 2	Diagonally	Slow	360	31.20
Camera 2	Left to Right	Medium	480	18.00
Camera 2	Front to back	Medium	360	24.00
Camera 2	Diagonally	Medium	480	18.00
Camera 2	Left to Right	Fast	480	24.00
Camera 2	Front to back	Fast	360	18.00
Camera 2	Diagonally	Fast	335	16.80

5.2 Number of Frames Value Determination for Recording

The purpose of this experiment was to determine an optional number of frames to be stored after a single classification. This experiment was done with two different size (pixel) cameras. One camera had high resolution, approximately 16 MP and the second was a 2 MP camera. In the experiment, a person is moved across a camera with different speeds e.g., slow, medium, and fast and its time and number of frames are recorded (Table 5). The optimal value of N found in this experiment is **270** number of frames and **23** seconds after taking the average of all frames and time taken.

5.3 Model Tuning

In the YOLOv5 model there is a tradeoff between the detection speed and accuracy of detected object. The purpose of this experiment is to find the opti-

mal detection speed for this system with minimum or no reduction in accuracy. Increasing the speed may result to save up to 80% of the time taken to detect objects at the cost of a slight reduction of accuracy. The detection speeds are flash, normal, fast, faster, and fastest. The observations are given in Table 6. We select the YOLOv5 model with a detection speed Fast for our application.

5.4 Baseline Comparison

We compare our solution (SVSS) with two baseline approaches 1) Continuous Object Detection (COD) approach - every frame is classified in real-time and 2) Motion Detection and Classification (MDC) approach - frames are classified only when motion is detected. In Table 7, we measure the CPU cost using the task manager of the local machine and the Python library Psutil [31].

Table 6. Comparison of Detection Speeds, Time, and Accuracy.

Detection Speed	Detection Time (s)	Accuracy
Normal	3.1	99.99%
Fast	1.8	90%
Faster	0.60	70%
Fastest	0.33	52.4%
Flash	0.2	25%

Table 7. Computational Cost Comparison.

System	Model	COD	MDC	SVS Sys
Task Manager (Minimum percentage)	Flash	15.1%	5.4%	3.1%
Task Manager (Average percentage)	Flash	42.9%	15.3%	9.9%
Task Manager (Maximum percentage)	Flash	57.6%	30.7%	29.1%
Psutil Library (Minimum percentage)	Flash	16%	10.3%	6.6%
Psutil Library (Average percentage)	Flash	35.8%	24%	15.6%
Psutil Library (Maximum percentage)	Flash	71%	24.4%	41.6%
Task Manager (Minimum percentage)	Normal	74.7%	22.1%	22.1%
Task Manager (Average percentage)	Normal	80.0%	43.5%	35.1%
Task Manager (Maximum percentage)	Normal	92.1%	81.2%	78.8%
Psutil Library (Minimum percentage)	Normal	31.6%	6.2%	5.8%
Psutil Library (Average percentage)	Normal	96.2%	26.6%	14.3%

We compare the CPU consumption for continuous fifty seconds and the results (Fig. 3) shows consistent improvement of the proposed model over the

baselines. The COD remains at peak all the time and MDC remains on peak more frequently. In contrast, the proposed system touches the peak rarely and operates on low CPU consumption most of the time.

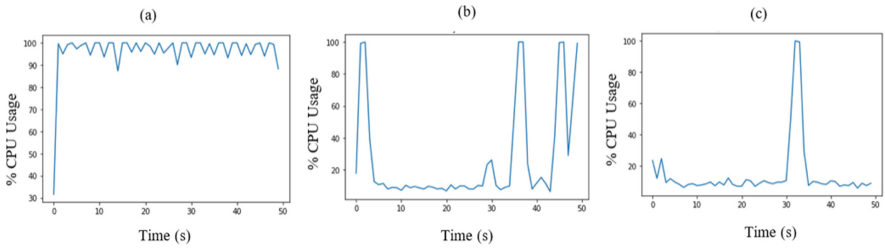

Fig. 3. CPU Consumption Graph for 50 s

6 Conclusion

The Smart Video Surveillance System (SVS System) is developed to overcome the issues and problems related to surveillance. As existing surveillance systems hold many issues like monitoring problems, issues in browsing and searching, cost of storage, no on-time notification, and high computation power consumption in case of continuous classification. The proposed system is one solution for all these issues. SVS System has five main components Motion Detection, Object Change Detection, Object Detection and Classification, Storage Service, and Notification Service. The motion Detection component is responsible for detecting motion in the captured frame. The object Change Detection component is responsible for detecting change, add or elimination of object(s) in consecutive frames. The object Detection and Classification component is responsible for object detection and predicting its label. The Storage Service is responsible for storing videos when specific conditions are met and maintaining information regarding each video recorded. Finally, Notification Service is responsible for sending notifications on the user's cell phone. The testing results show its performance in real scenarios. The optimizations in the algorithm have improved the overall system's performance using features like Motion Detection, Change Object Detection and Object Detection and Classification. In addition to this, it reduces the painful efforts of searching for a particular object in the library of recorded videos by using optimizations in video storage and notification service. Finally, the presented SVS System offers unique features in searching for a particular object and on-time notification on targeted object detection.

References

1. DVR Storage Calculator for Analog Security Cameras. Supercircuits (2020), https://www.supercircuits.com/resources/tools/security-dvr-storage-calculator, Accessed 20 Mar 2023

2. Ahn, H., Cho, H.-J.: Research of multi-object detection and tracking using machine learning based on knowledge for video surveillance system. Pers. Ubiquit. Comput. **26**(2), 385–394 (2019). https://doi.org/10.1007/s00779-019-01296-z
3. Alsmirat, M., Sarhan, N.J.: Intelligent optimization for automated video surveillance at the edge: a cross-layer approach. Simul. Model. Pract. Theory **105**, 102171 (2020)
4. Basavaraj, G., Kusagur, A.: Vision based surveillance system for detection of human fall. In: RTEICT, pp. 1516–1520. IEEE (2017)
5. Bradski, G.: The OpenCV Library. Dr. Dobb's Journal of Software Tools (2000)
6. Chandana, S.: Real time video surveillance system using motion detection. In: 2011 Annual IEEE India Conference, pp. 1–6. IEEE (2011)
7. Chitra, M., Geetha, M.K., Menaka, L.: Occlusion detection in visual scene using histogram of oriented gradients. In: ICEVENT, pp. 1–5. IEEE (2013)
8. Cob-Parro, A.C., Losada-Gutiérrez, C., Marrón-Romera, M., Gardel-Vicente, A., Bravo-Muñoz, I.: Smart video surveillance system based on edge computing. Sensors **21**(9), 2958 (2021)
9. Daigavane, P., Bajaj, P.R., Daigavane, M.: Vehicle detection and neural network application for vehicle classification. In: 2011 International Conference on Computational Intelligence and Communication Networks, pp. 758–762. IEEE (2011)
10. Dedeoğlu, Y.: Moving object detection, tracking and classification for smart video surveillance. Ph.D. thesis, Bilkent Universitesi (Turkey) (2004)
11. Fang, L., Meng, Z., Chen, C., Hui, Q.: Smart motion detection surveillance system. In: International Conference on Education Technology and Computer, pp. 171–175. IEEE (2009)
12. Fang, Y., Guo, X., Chen, K., Zhou, Z., Ye, Q.: Accurate and automated detection of surface knots on sawn timbers using YOLO-V5 model. BioResources **16**(3), 5390 (2021)
13. Feiran, F., Ming, F., Huamin, Y.: Temporal difference method based on positive and negative energy distribution in moving objects detection. In: IST, pp. 1–5. IEEE (2017)
14. Ji, P., Kim, Y., Yang, Y., Kim, Y.S.: Face occlusion detection using skin color ratio and LBP features for intelligent video surveillance systems. In: FedCSIS, pp. 253–259. IEEE (2016)
15. Jocher, G., et al.: ultralytics/yolov5: v3.1 - Bug Fixes and Performance Improvements (2020). https://doi.org/10.5281/zenodo.4154370
16. Ju, J., Ku, B., Kim, D., Song, T., Han, D.K., Ko, H.: Online multi-person tracking for intelligent video surveillance systems. In: ICCE, pp. 345–346. IEEE (2015)
17. Kalirajan, K., Sudha, M.: Moving object detection for video surveillance. Sci. World J. **2015** (2015)
18. Kim, K., Davis, L.S.: Object detection and tracking for intelligent video surveillance, pp. 265–288. Multimedia Analysis, Processing and Communications pp (2011)
19. Kruegle, H.: CCTV Surveillance: Video practices and technology. Elsevier (2011)
20. Liu, W., Liao, S., Hu, W.: Perceiving motion from dynamic memory for vehicle detection in surveillance videos. IEEE Trans. Circuits Syst. Video Technol. **29**(12), 3558–3567 (2019)
21. Liu, Y., Li, Z., Xiong, H., Gao, X., Wu, J.: Understanding of internal clustering validation measures. In: ICDM, pp. 911–916. IEEE (2010)
22. Luo, H., Liu, J., Fang, W., Love, P.E., Yu, Q., Lu, Z.: Real-time smart video surveillance to manage safety: a case study of a transport mega-project. Adv. Eng. Inform. **45**, 101100 (2020)

23. Miao, Z., Zou, S., Li, Y., Zhang, X., Wang, J., He, M.: Intelligent video surveillance system based on moving object detection and tracking. DEStech Trans. Eng. Technol. Res. **11**, 2016 (2016)
24. Mishra, A.A., Srinivasa, G.: Automated detection of fighting styles using localized action features. In: ICISC. pp. 1385–1389. IEEE (2018)
25. Nikouei, S.Y., Xu, R., Nagothu, D., Chen, Y., Aved, A., Blasch, E.: Real-time index authentication for event-oriented surveillance video query using blockchain. In: ISC2, pp. 1–8. IEEE (2018)
26. Paul: Video Resolution VS. Frames Per Second. Thinpig-media (2019), https://thinpigmedia.com/blog/decisions-decisions-video-resolution-vs-frames-per-second
27. Pramanik, A., Sarkar, S., Maiti, J.: A real-time video surveillance system for traffic pre-events detection. Accid. Anal. Prev. **154**, 106019 (2021)
28. Qureshi, F.Z., Terzopoulos, D.: Surveillance in virtual reality: System design and multi-camera control. In: CVPR, pp. 1–8. IEEE (2007)
29. Rai, M., Husain, A.A., Maity, T., Yadav, R.K., Neves, A.: Advance intelligent video surveillance system (AIVSS): a future aspect. Intell. Video Surveill. **37** (2019)
30. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: CVPR, pp. 779–788 (2016)
31. Rodola, G.: Psutil package: a cross-platform library for retrieving information on running processes and system utilization. Google Scholar (2016)
32. Salahat, E., Saleh, H., Mohammad, B., Al-Qutayri, M., Sluzek, A., Ismail, M.: Automated real-time video surveillance algorithms for SOC implementation: A survey. In: ICECS, pp. 82–83. IEEE (2013)
33. Sase, P.S., Bhandari, S.H.: Human fall detection using depth videos. In: SPIN, pp. 546–549. IEEE (2018)
34. Shafie, A., Ali, M., Hafiz, F., Ali, R.M.: Smart video surveillance system for vehicle detection and traffic flow control. J. Eng. Sci. Technol. **6**(4), 469–480 (2011)
35. Singla, N.: Motion detection based on frame difference method. Int. J. Inf. Comput. Technol. **4**(15), 1559–1565 (2014)
36. SuganyaDevi, K., Malmurugan, N., Manikandan, M.: Object motion detection in video frames using background frame matching. Int. J. Comput. Trends Technol **4**, 1928–1931 (2013)
37. Tuan, M.C., Chen, S.L.: Fully pipelined VLSI architecture of a real-time block-based object detector for intelligent video surveillance systems. In: ICIS, pp. 149–154. IEEE (2015)
38. Wang, R., Tsai, W.T., He, J., Liu, C., Li, Q., Deng, E.: A video surveillance system based on permissioned blockchains and edge computing. In: BigComp, pp. 1–6. IEEE (2019)
39. Wang, Z., Zhao, Y., Zhang, J., Guo, Y.: Research on motion detection of video surveillance system. In: 2010 3rd International Congress on Image and Signal Processing, vol. 1, pp. 193–197. IEEE (2010)
40. Xu, J.: A deep learning approach to building an intelligent video surveillance system. Multimedia Tool Appl. **80**(4), 5495–5515 (2021)
41. Yoon, C.S., Jung, H.S., Park, J.W., Lee, H.G., Yun, C.H., Lee, Y.W.: A cloud-based utopia smart video surveillance system for smart cities. Appl. Sci. **10**(18), 6572 (2020)
42. Zhang, S., et al.: Pedestrian search in surveillance videos by learning discriminative deep features. Neurocomputing **283**, 120–128 (2018)
43. Zhou, Z., Yu, H., Shi, H.: Optimization of wireless video surveillance system for smart campus based on internet of things. IEEE Access **8**, 136434–136448 (2020)