IRSTI 28.23.29 https://doi.org/10.26577/jpcsit2023v1i4a7

N. Azatbekuly* , A. Mukhanbet

Al-Farabi Kazakh National University, Almaty, Kazakhstan
*e-mail: nurtugang17@gmail.com

OPTIMIZED INTELLIGENT MODULES FOR VIDEO SURVEILLANCE AND MONITORING

Abstract. Platform of optimized intelligent modules, designed for intelligent video surveillance and control systems, is an outstanding tool for implementing effective security measures and improving monitoring processes. This paper examines the development and optimization of several important modules, including a facial recognition system, line intersection detection algorithms, methods for detecting the presence of a person in specified zones and an algorithm for searching for people. The most productive algorithms, such as Faster R-CNN (Faster Region-based Convolutional Neural Network), YOLO (You Only Look Once), SSD (Single-Shot Detector), were researched, thereby work was carried out to improve and optimize the YOLO algorithm. In addition to the use of this algorithm, optimizations were carried out, including image processing methods (including scaling) and a frame skipping mechanism using parallel computing, which significantly reduced the computational load. The resulting platform provides users with the ability to effectively monitor and analyze video streams, automatically identify potential threats and events, which makes it the optimal solution for ensuring security in a variety of applications, including public places, enterprises and critical infrastructure facilities. The results of this paper provide new prospects for improving video surveillance and control systems, contributing to an increase in the level of security and efficiency of actions.

Key words: Intelligent Surveillance, CNN, YOLO, Optimization, Multithreading.

1. Introduction

The development of an intelligent video surveillance and control system that has the ability to function effectively with limited computing resources is an urgent paper in modern information technology. The present time is characterized by an increase in the volume of video data and growing requirements for the performance of video surveillance systems, which is dictated by both the sphere of public security and corporate interests. It was found that the creation of a system that does not resort to excessive computing resources, considering modern technological requirements, is a challenge. It is important to take into account that such a system should ensure stable operation, free from failures and errors.

There are several papers on a similar topic. For example, in [1] good results have been achieved using the high-level TensorFlow library, in [2] there is good information about main issues related to the intelligent monitoring systems used in public spaces.

Based on the complexity of the task, in this paper we pay attention to the optimization of algorithms and models of computer vision by manipulating images. This technique is designed to reduce the load on computing resources while maintaining the accuracy of the system. In addition, an important stage in the development of such systems is the use of frame skipping techniques, which contributes to the rational use of computing power. This paper offers a detailed analysis of the process of developing and optimizing an intelligent video surveillance and control system, highlighting the key aspects, methodology and achievements that allowed us to solve complex problems and create a system that combines high efficiency and optimal use of resources. The problems discussed in this paper are partially described in [3].

2. Methodology

This part will show our research and opinions on our topic in the context of the developed optimized modules, coupled with the theory and justification of the methods themselves.

2.1. Justification of the choice of the YOLO algorithm

YOLO (You Only Look Once), being a super-high-level tool, is not deprived of poor quality of work or inefficient use of resources [4]. Surface

studies and tests of models built on PyTorch or Tensorflow frameworks were carried out, the results and quality of which were much inferior to the ready-made YOLO object detection models [5]. Thus, this cluster of ready-made models is suitable for using them in production. But the existing problems have not gone away, its customized solutions are given in this paper.

The task was mainly based on the optimization of human detection, so other algorithms such as Faster R-CNN (Faster Region-based Convolutional Neural Network) [6] and SSD (Single-Shot Detector) were not used, due to a significantly lower detection rate than the YOLO algorithm (an algorithm with a single-stage approach in architecture). For detection in almost all cases in the environment of our developed modules, the accuracy of the fast YOLO algorithm of the last eighth version turned out to be quite enough. As befits any convolutional neural network, YOLO has the same 2 main stages: feature extraction and detecting, in a single-stage form [7]. Here are the main points highlighted in the course of the research in the YOLO architecture (see Figure 1):

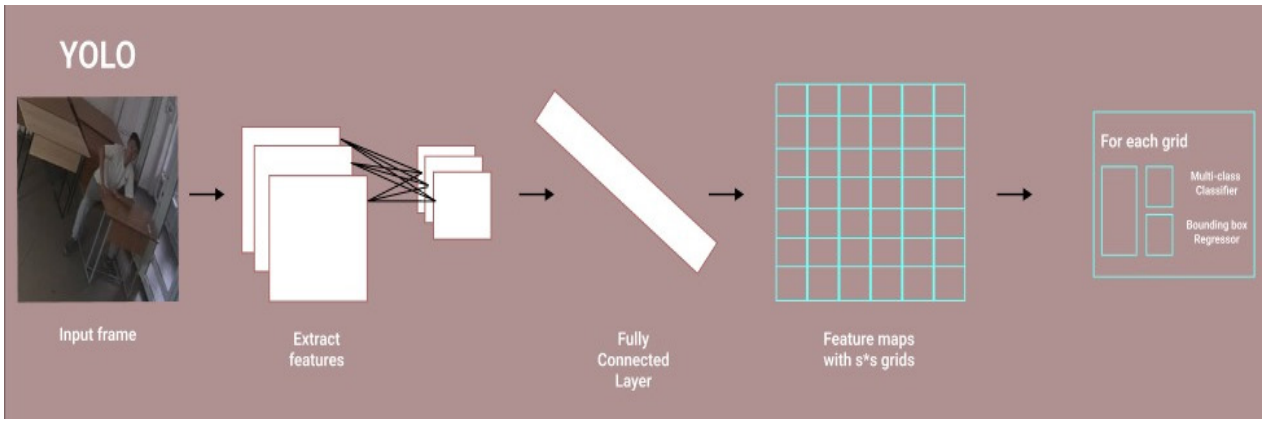

Figure 1 – Progress of post-traumatic stress disorder of personality

2.2. Optimization with multithreading

Complex and resource-intensive tasks of working with a video stream can also be solved live due to multithreading. The methodology for optimizing deep-learning algorithms using a parallel approach and potential directions for parallelism in deep learning are described in [8]. Work in this paper describes a solution that is a simple implementation for the capture and processing of video frames in a separate stream, with callback functions and blocking and non-blocking frame extraction options. In particular, the following aspects of the implementation and application of multithreading in working with video streams and processing it with the YOLO framework should be considered (see Figure 2).

For efficient frame processing in the YOLO algorithm, the "FreshestFrame" class was developed using multithreading. Consider the key elements of its implementation include:

1. Initialization. When creating an instance of the class, the variables necessary for working with video capture and multithreading are set. A blocking condition ("self.cond") is supported, which ensures waiting for a new frame to appear.

2. Starting A Stream. When the stream starts ("self.start()"), an infinite loop of reading frames from the video capture begins. Each frame is published using a lock condition, which allows other threads to wait for a fresh frame to appear.

3. Reading and Processing Frames. In the "run()" method, frames are read from the video capture, and their state variables are updated. If there is a callback ("self.callback"), the corresponding frame processing is performed.

4. Reading Frames from the Main Stream. The "read()" method allows the main thread to read frames. When using the lock condition, it is blocked until a new frame appears. You can set a sequential frame number ("seqnumber') to lock until the specified number is reached.

5. Stop and Release. When the "release()" method is called, the stream is stopped and video capture resources are released.

6. Main Stream. The main thread continues to perform other tasks without being blocked while waiting for new frames.

The use of multithreading in the "FreshestFrame" class ensures efficient and continuous processing of the video stream, which is especially important for the YOLO algorithm, which requires fast analysis of multiple frames. The above optimization using multithreading gives the following results in the development of intelligent modules for video surveillance (see Figure 3):

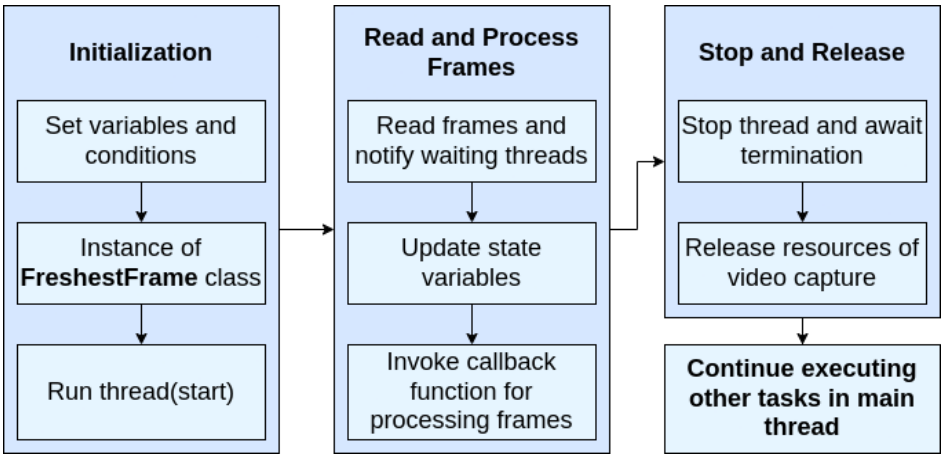

Figure 2 – Schematic representation of multithreading routine on reading and processing frames

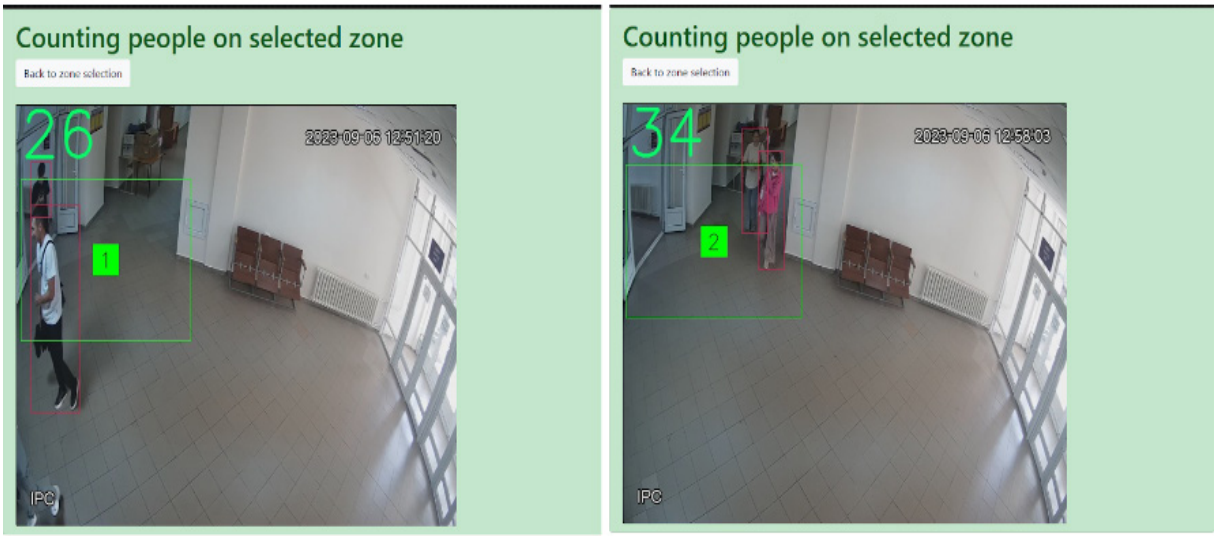

Figure 3 – FPS improvement after optimization with multithreading

2.3. Optimization with image manipulation

Image manipulation techniques play a critical role in optimizing and improving the performance of an integrated video surveillance and control system [9] [10]. In this article, image manipulation techniques have been developed and applied, aimed at improving the accuracy and reliability of video stream analysis in a variety of conditions. The use of these methods has had a significant impact on the effectiveness of the system. Let's look at a few key methods of image manipulation:

Increasing the size of detected objects: In cases when the objects on the video stream are small or their image is not clear enough, a method of increasing the size of detected objects has been implemented. This method allows you to more accurately extract

features and provide more stable recognition, even with limited visibility.

Changing contrast and brightness: Analyzing a video stream in different light conditions can be a challenge. To solve this problem, methods of changing the contrast and brightness of images were applied. This helped to better highlight objects in the background and improve overall clarity.

Noise Filtering: Video data may contain noise and artifacts that may affect detection accuracy. Filtering techniques such as median filter or Gaussian filter have been applied to reduce the effect of noise to improve image quality.

One example of the use of frame manipulation methods is the face recognition module, in which all the listed techniques were applied. Before direct face recognition, a person is detected by one of the YOLOv8 detection models, then the frame with the person is enlarged and the listed filters are applied to restore the enlarged and fuzzy image. After all these manipulations, a face will be recognized using face_recognition (see Figure 4).

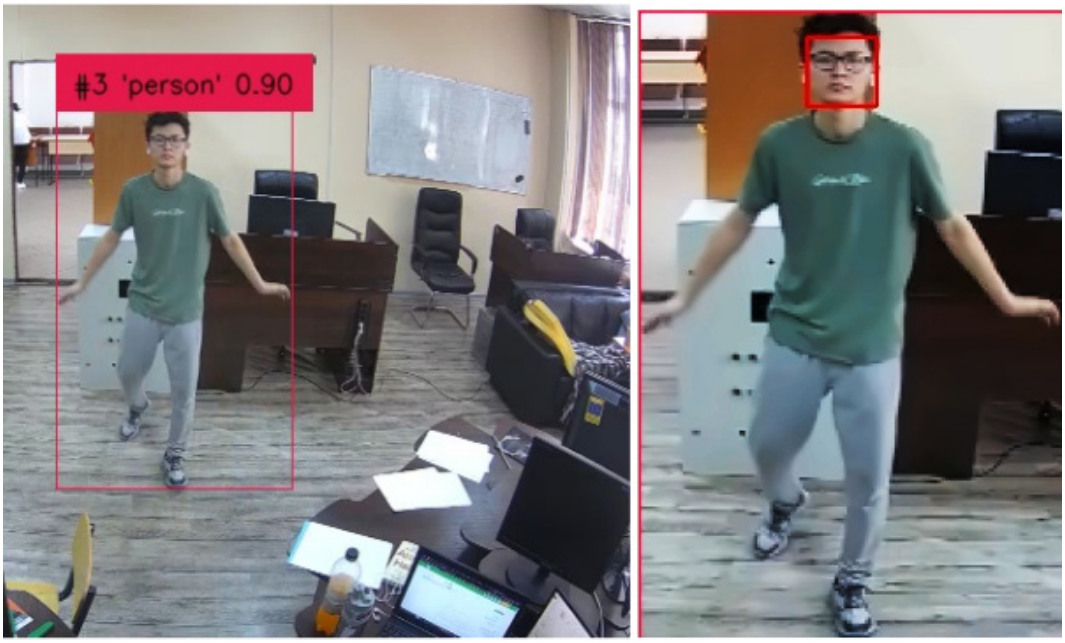

Figure 4 – Face detecting improvement after frame optimization

3. Performance evaluation

In this part, in order to visualize the optimized results obtained, FPS indicators were compared, as well as the accuracy of the modules was visually checked.

3.1. Performance comparison

Coupled with the included local Django server and other manipulations with the video stream that was captured by the open-cv module, which gradually but not significantly reduce the frames per second, several tests were conducted on the performance of the models (see Figures 4, 5). But it should also be noted that although the provided models from the YOLO – versions tiny, small, medium, large, extra-large, promise to be more accurate as the model becomes more complex (increasing the number of layers and, accordingly, parameters), these tests show slightly unexpected results, at least for human recognition. This applies to medium and large models, which did not justify the promised high-performance indicator in practice, being approximately on par with the small model. And of course, the fact remains true that the value of frames per second varies depending on the workload of processing the video stream (specifically on the number of people). The following tests were made on the basis of the NVIDIA GeForce GTX 1660 SUPER GPU with 6GB dedicated RAM, with 16 GB RAM. The video stream from the camera with a resolution of 640 by 544 (348160 pixels) was processed.

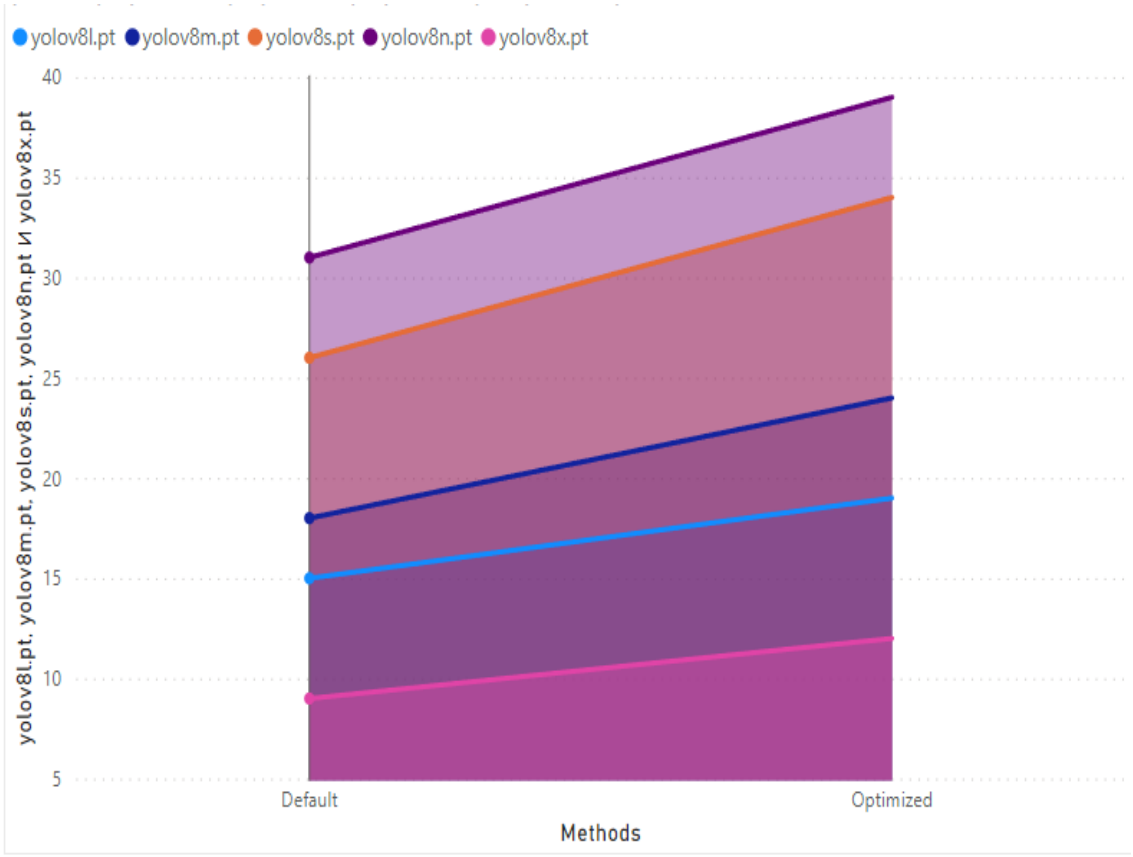

Figure 5 – Comparison of FPS rate of default and optimized versions

3.2. Comparison of detection quality

The face_recognition library does not allow to recognize, even detect faces on the camera at decent distances, say a few meters. But with the help of image manipulation methods, as well as playing with hyperparameters, in particular the threshold value, a good result was achieved in face recognition. The previous images (see Figure 3) show an enlarged, filtered, optimized frame with a person where a person's face is clearly detected. At the level of our development, all these points are registered in the backend and are not visible to the user of the developed software.

4. Results

As a result, this article applies the results of the work of the existing development, previously optimized by the methods described in part of the methodology. In connection with the results achieved in the development of optimized modules for detecting and recognizing faces, monitoring people in the zone, monitoring the intersection of the line, detecting abandoned objects and searching for a person, the possibility of adding triggers and viewing analytics were also added to the existing developments. The correct operation of powerful YOLO models makes it possible to add secondary features of the developed system, such as sending messages to messengers or to e-mail according to a given trigger (an event created by the software user). A smart video surveillance and monitoring system was implemented using computer vision algorithms, and accordingly, the methods proposed in the article were used to optimize them. Figure 6 shows screenshots of the operation of the module for monitoring people in the zone.

The human line crossing module is also a resource-intensive task, since in the context of frame processing using YOLO for the human detection subtask, an additional task of calculating coordinates in each frame is added, these tasks have been optimized using the proposed methodology. Screenshots of the implementation of this module are presented in Figure 7.

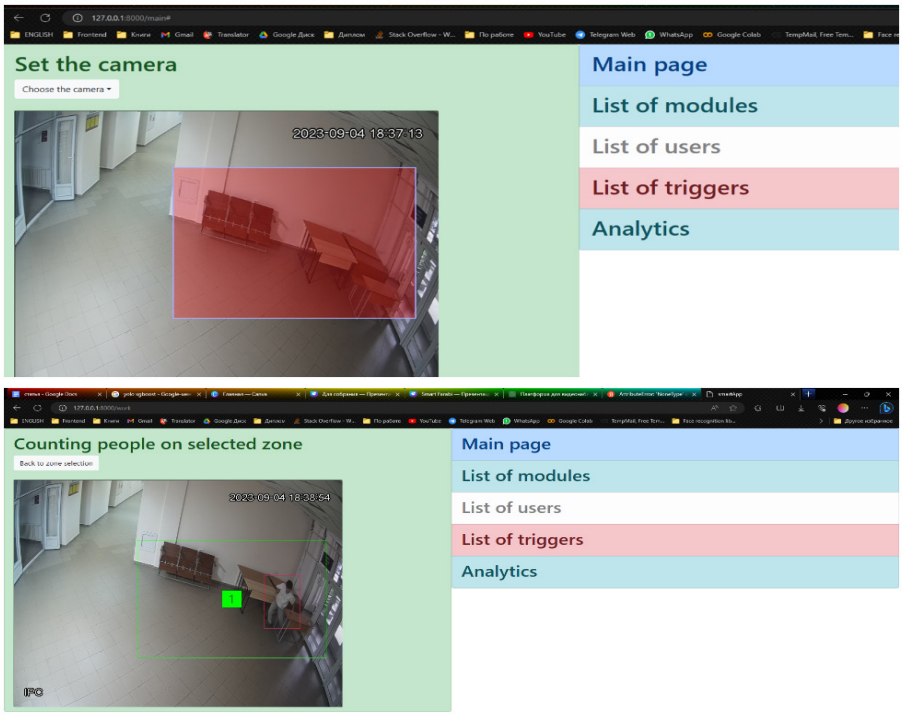

Figure 6 – Screenshot of the results of the development of the zone detection module

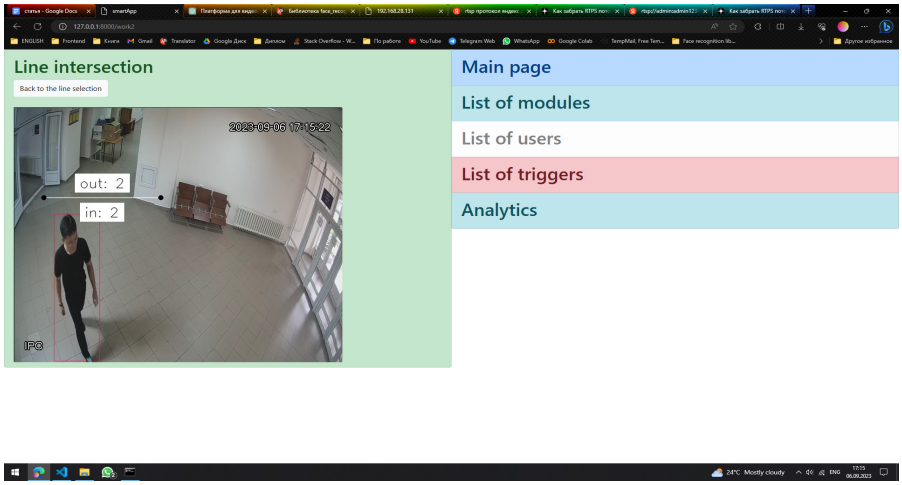

Figure 7 – Screenshot of the results of the development of the line intersection module

People searching module (see Figure 8) is the most high-performance in the existing development, because it includes the tasks of human detection as well as facial recognition. During its implementation, before detecting a face, the process of detecting a person and subsequent image manipulations, such as zooming, contrast enhancement, and noise removal, are pre-launched. Additionally, in addition to the proposed multithreading technique, the yolo ono model was used to further improve performance.

Figure 8 – Screenshot of the results of the development of the human search module

5 Conclusion

In this research paper, the results of the development and optimization of modules for an integrated video surveillance and control system based on YOLO algorithm were presented. Based on a number of methods and techniques, including multithreading and image manipulation, there has been a significant improvement in the performance and accuracy of video stream analysis.

The system presented in this paper has the ability to detect and classify objects in real time, which significantly increases its potential application in the field of security, monitoring and automation of processes. Key modules, including face recognition, object detection, and counting and motion tracking, have demonstrated outstanding performance and accuracy under various operating conditions.

The results of the comparisons indicate a significant increase in performance, as well as maintaining high accuracy of object detection. It is worth noting the increase in frame rate (FPS), which is a key indicator for real-time system.

In conclusion, this paper highlights the importance of a combined approach, including both technical optimizations and image manipulation methods, to create highly efficient integrated video surveillance and control systems. The results obtained have the potential for wide application in various fields, including ensuring the safety of society and industrial applications.

References

1. Kadikonda Mounika, Vancha Vaishnavi Reddy, Asma Begum (2021). “Intelligent video surveillance using deep learning”. International Journal for Innovative Engineering and Management Research. Volume 11 Issue no.06, (2021): 566-579.
2. Wang, Xiaogang. 2013. “Intelligent Multi-Camera Video Surveillance: A Review.” Pattern Recognition Letters. Elsevier BV. https://doi.org/10.1016/j.patrec.2012.07.005.
3. Liu, Li, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. 2019. “Deep Learning for Generic Object Detection: A Survey.” International Journal of Computer Vision. Springer Science and Business Media LLC. https://doi.org/10.1007/s11263-019-01247-4.
4. Rekha B. S., Athiya Marium, Dr. G. N. Srinivasan, Supreetha A. Shetty. 2020. “Literature Survey on Object Detection using YOLO”. International Research Journal of Engineering and Technology (IRJET). Volume 07, Issue no. 06 (2020): 3082-3088.
5. Redmon, Joseph, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. “You Only Look Once: Unified, Real-Time Object Detection.” 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE. https://doi.org/10.1109/cvpr.2016.91.
6. Girshick, Ross. 2015. “Fast R-CNN.” arXiv. https://doi.org/10.48550/ARXIV.1504.08083.
7. Jiang, Peiyuan, Daji Ergu, Fangyao Liu, Ying Cai, and Bo Ma. 2022. “A Review of Yolo Algorithm Developments.” Procedia Computer Science. Elsevier BV. https://doi.org/10.1016/j.procs.2022.01.135
8. Ben-Nun, Tal, and Torsten Hoefler. 2019. “Demystifying Parallel and Distributed Deep Learning.” ACM Computing Surveys. Association for Computing Machinery (ACM). https://doi.org/10.1145/3320060.
9. Lou, Haitong, Xuehu Duan, Junmei Guo, Haiying Liu, Jason Gu, Lingyun Bi, and Haonan Chen. 2023. “DC-YOLOv8: Small Size Object Detection Algorithm Based on Camera Sensor.” MDPI AG. https://doi.org/10.20944/preprints202304.0124.v1
10. Wei, Xiaoyan, Yirong Wu, Fangmin Dong, Jun Zhang, and Shuifa Sun. 2019. “Developing an Image Manipulation Detection Algorithm Based on Edge Detection and Faster R-CNN.” Symmetry. MDPI AG. https://doi.org/10.3390/sym11101223.