DOI: 10.65322/j.srd.04182026.26

EFFICIENT OBJECT DETECTION USING A NOVEL FRAME COMPARISON METHOD WITH APPLICATION IN INTELLIGENT TRANSPORTATION SYSTEMS

Begmatov Shohruh, *Doctoral Researcher,*
Arabboev Mukhriddin, *Doctoral Researcher,*
Nishanov Akhram, *Professor*
Tashkent University of Information Technologies named after Muhammad al-Khwarizmi

Abstract: *This paper presents a novel approach to improving object detection in intelligent transportation systems (ITS) through the use of a frame comparison method, designed to enhance computation-al efficiency while maintaining high detection accuracy. Traditional object detection techniques often process every frame in a video stream, leading to high computational costs, latency, and energy inefficiency, particularly in resource-constrained environments. The proposed frame comparison method strategically processes only selected frames that exhibit significant changes, thereby reducing redundant computations and accelerating processing times. Experimental re-sults indicate that the proposed method significantly surpasses the traditional SSIM-based ap-proach in terms of processing speed and resource efficiency, while maintaining comparable ob-ject detection accuracy. These findings suggest that the frame comparison method can serve as a valuable tool for enhancing the scalability and responsiveness of ITS applications.*

Annotatsiya: *Ushbu maqolada intellektual transport tizimlarida (ITS) obyektlarni aniqlash samaradorligini oshirishga qaratilgan, hisoblash samaradorligini yaxshilash va ayni paytda yuqori aniqlikni saqlab qolish imkonini beruvchi yangi kadrlarni taqqoslash usuli taklif etiladi. Obyektlarni aniqlashning an'anaviy usullarida ko'pincha videooqimdagi har bir kadr qayta ishlanadi, bu esa, ayniqsa, hisoblash resurslari cheklangan muhitlarda yuqori hisoblash xarajatlari, kechikishlar va energiya sarfining ortishiga olib keladi. Taklif etilayotgan kadrlarni taqqoslash usuli sezilarli o'zgarishlar kuzatilgan kadrlarni tanlab qayta ishlashga asoslangan bo'lib, ortiqcha hisoblashlarni kamaytirish va qayta ishlash tezligini oshirish imkonini beradi. Eksperimental natijalar shuni ko'rsatadiki, taklif etilgan usul qayta ishlash tezligi va hisoblash resurslaridan foydalanish samaradorligi bo'yicha an'anaviy SSIM asosidagi yondashuvdan sezilarli darajada ustun bo'lib, obyektlarni aniqlash aniqligini taqqoslanadigan darajada saqlab qoladi. Olingan natijalar kadrlarni taqqoslash usuli intellektual transport tizimlari ilovalarining masshtablanuvchanligi va tezkor javob berish xususiyatlarini oshirish uchun samarali vosita bo'lib xizmat qilishi mumkinligini ko'rsatadi.*

Аннотация: *В данной статье представлен новый подход к повышению эффективности обнаружения объектов в интеллектуальных транспортных системах на основе метода сравнения кадров, разработанного для повышения вычислительной эффективности при сохранении высокой точности обнаружения. Традиционные методы обнаружения объектов зачастую предусматривают обработку каждого кадра видеопотока, что приводит к высоким вычислительным затратам, задержкам и неэффективному энергопотреблению, особенно в условиях ограниченных вычислительных ресурсов. Предлагаемый метод сравнения кадров предусматривает выборочную обработку только тех кадров, в которых наблюдаются существенные изменения, что позволяет сократить избыточные вычисления и повысить скорость обработки. Экспериментальные результаты показывают, что предложенный метод значительно превосходит традиционный подход на основе SSIM по скорости обработки и эффективности использования вычислительных ресурсов, сохраняя при этом сопоставимую точность обнаружения объектов. Полученные результаты свидетельствуют о том, что метод сравнения кадров может служить эффективным*

DOI: 10.65322/j.srd.04182026.26

инструментом для повышения масштабируемости и оперативности приложений интеллектуальных транспортных систем.

Keywords: *real-time object detection; frame-comparison; smart vision.*

Kalit so'zlar: *real vaqt rejimida obyektlarni aniqlash; kadrlarni taqqoslash; aqlli ko'rish tizimlari.*

Ключевые слова: *Обнаружение объектов в реальном времени; сравнение кадров; интеллектуальные системы машинного зрения.*

1. Introduction

Computer vision is a branch of artificial intelligence that empowers computers to comprehend and interpret visual data and is a cornerstone of intelligent transportation systems (ITS) [1,2]. Due to rapid developments in the field of ITS, object detection plays a crucial role in ensuring the safety, efficiency, and reliability of modern transportation networks. As vehicles and infrastructure become increasingly equipped with sensors and cameras[3], the ability to accurately detect and track objects such as pedestrians [4-6], vehicles [7-16] and obstacles [3], [17,18] in real-time is more important than ever. However, traditional object detection methods often face significant challenges, particularly when deployed in resource-constrained environments where real-time performance is essential [19, 20]. The high computational demands of processing every frame in a video stream can lead to latency, energy inefficiency, and decreased overall system performance. This paper addresses these challenges by introducing an innovative approach that balances accuracy and efficiency.

The core idea of the frame comparison method is to selectively process only certain frames within a video sequence, thereby reducing the computational load without sacrificing detection accuracy. Unlike conventional methods that analyze every frame, this approach strategically skips frames where minimal or no significant changes are detected. This not only speeds up the processing time but also conserves computational resources, making it an ideal solution for real-time applications in ITS, such as autonomous driving, traffic monitoring, and collision avoidance systems. By focusing on frames that are more likely to contain new or relevant information, the proposed method ensures that critical objects are still detected in a timely manner, while redundant computations are minimized.

One of the key advantages of the frame comparison method is its ability to maintain robust detection performance. The proposed method addresses these challenges by incorporating algorithms that dynamically adjust the frame comparison criteria based on the context and movement within the scene. This ability ensures that the system remains responsive and accurate in highway traffic scenarios.

The paper further explores the implementation details of the frame comparison method, including the selection criteria for frames, the integration with existing object detection models, and the impact on system performance. Extensive experiments are conducted to evaluate the effectiveness of the proposed method. The results demonstrate the proposed method outperformed the SSIM-based method with significant improvements in processing speed and resource efficiency, without compromising object detection accuracy. These findings suggest that the frame comparison method could be a valuable addition to the toolkit of techniques used in ITS, enabling more scalable and responsive systems.

The main contribution of this work is three-fold.

- A novel frame comparison method is proposed, which introduces real-time object detection for ITS. The method significantly improves the performance of object detection in highway images.

DOI: 10.65322/j.srd.04182026.26

- The frame comparison method is designed to address the challenges of real-time object detection in resource-constrained environments, such as autonomous driving and traffic monitoring.
- The paper describes the creation of new dataset specifically designed for the In this section, we give a concise review of the traditional AI-based object detection model (see Figure 1). In addition, we introduce an existing SSIM-based method for real-time object detection (see Figure 4(a)). Moreover, we propose a novel frame comparison method (see Figure 4(b)) to significantly improve the performance of real-time object detection while minimizing overhead.
- imple-mentation and evaluation of the proposed frame comparison method.

The rest of this paper is organized as follows. Section 2 introduces related works about frame filtering/skipping methods used for object detection. Section 3 describes the proposed method in detail. Section 4 explains the dataset and model used in this work, and the implementation of the proposed method. The experimental results are presented in Section 5. Finally, conclusions are given in Section 6.

2. Related Work

In recent years, numerous studies have focused on the development of object detection models using frame filtering/skipping for various applications, including ITS.

In [21], frame skipping method is used for speeding up multi object tracking. This research combined a slow-accurate deep-learning based detection method with a fast-moderately accurate tracking method. The main contribution of the proposed approach is to improve the speed of the Multiple Object Tracking (MOT) system while maintaining reasonable accuracy.

In [22], the authors proposed optimization techniques to extend the use of YOLOv3 tiny to accurately detect small, fast-moving objects. They used a frame-skipping method to ensure real-time application.

In [23], a novel light- weight object detector named FDet to improve the efficiency of query system is introduced. The system initially filters out static frames. Then, the FDet, a fast object detector, analyzes the remaining frames. The FDet uses keypoints and corners to efficiently predict bounding boxes, avoiding complex anchor box calculations.

In [24], a method to simplify object detection in video streams is proposed. For each frame, the system compares it to the previous frame using temporal subtraction and Oriented FAST and Rotated BRIEF (ORB) feature matching. Based on these comparisons, it decides whether to run YOLOv3 or reuse the previous frame's detection results.

In [25], a novel approach for multi-object dynamic frame detection and skipping using Deep Learning is presented. The proposed model simplifies the framework by skipping similar frames and avoiding redundant classifications. To skip frames, we compare their structural similarity. Frames with over 90% similarity to previous frames are skipped to reduce RCNN workload.

The main contribution of this paper [26] is to address the challenges of real-time object detection using YOLO on resource-constrained embedded systems, particularly in the context of network cameras. They propose a novel architecture called YOLO with Adaptive Frame Control (AFC) to improve efficiency and real-time performance.

In [27], a model-agnostic method that uses frame-dropping to scale single-frame object detection models in tracking-by-detection systems is introduced. This approach dynamically adapts autonomous vehicle (AV) perception based on situational needs, reducing processing demands and energy consumption. It optimizes the balance between detection performance and resource efficiency in AVs.

In [28], a video object detection approach that enhances accuracy, efficiency, and reduces annotation costs is introduced. The proposed method improves accuracy by predicting object motion from static images using the law of common fate. Efficiency is achieved by processing

DOI: 10.65322/j.srd.04182026.26

only keyframes and predicting motion for intervening frames, reducing computational load. Annotation costs are minimized by annotating only sampled keyframes.

The main contribution of this paper [29] is the development of fast and efficient video summarization algorithms, specifically the hopscotch and lens algorithms, designed for security camera recordings on low-power devices like embedded systems. These algorithms were tested in an experimental environment with various scenarios, demonstrating their potential for rapid video processing.

In [30], a frame-skipping method for correlation filter-based visual object trackers, aimed at improving efficiency with minimal or no loss of accuracy is introduced. The method includes an adaptive tracking-skipping algorithm and Gaussian-based movement prediction, allowing the system to predict the position of objects in the next frame based on their movement in previous frames and determine whether the tracking process can be skipped.

In [31], the authors proposed a lightweight filtering method called L-filter, which accurately predicts empty video frames containing no objects of interest using hybrid time series analysis. By dropping these empty frames and focusing object detection only on nonempty frames, L-filter significantly improves the frame processing rate and scalability of real-time object detection.

Overall, the existing object detection methods mentioned in this section [21]–[31] emphasize that real-time object detection methods for ITS typically require substantial computing power and processing time. However, none of these contributions/studies explores the specific concept of frame comparison, a novel approach introduced in this paper to enable real-time object detection with lower computing resource consumption. By selectively skipping frames that lack relevant content, we significantly enhance processing speed and enable scalable real-time detection for multiple concurrent video streams.

3. The Proposed Method

This section describes the dataset, preprocessing procedure, proposed low-resource spatial-semantic feature extractor, embedding generation, similarity computation, and metric-learning objective used for person re-identification in crowded video surveillance.

3.1. Traditional AI based object detection

AI has transformed object detection by utilizing advanced algorithms and extensive datasets to accurately identify and locate objects in images or videos. Some of the main advantages of using AI for object detection are:

High Accuracy: AI models can achieve extremely high accuracy rates in object detection tasks, often surpassing human performance. This is because they can learn complex pat-terns and features from large amounts of data.

Real-time Processing: With improvements in hardware and optimization techniques, AI models can now process images and videos in real-time. This makes them suitable for applications such as autonomous vehicles, surveillance systems, and augmented reality.

Versatility: AI-based object detection algorithms can be used in a wide range of domains. They can detect objects of different sizes, shapes, and orientations, making them adaptable to various use cases.

AI presents significant benefits for object detection but also encounters several challenges that need to be addressed for widespread adoption.

Data Requirement: One of the main challenges in training accurate AI models is the necessity for large amounts of high-quality labeled data. Collecting and annotating such datasets can be a time-consuming and expensive process, especially for tasks that require domain-specific expertise.

Occlusion and Clutter: Handling occlusions and clutter in images remains a challenging problem for AI-based object detection. When objects are partially obscured or surrounded by other

DOI: 10.65322/j.srd.04182026.26

objects, accurately identifying and locating them can be difficult for the model. Developing robust algorithms to address these challenges is an ongoing area of research.

Computational Cost: Deep learning models, commonly used for object detection, can be computationally intensive, requiring powerful hardware and significant processing time. This can limit their use in resource-constrained environments, such as mobile devices or embedded systems.

In this paper, the main focus is on the issue of computational efficiency/optimization, one of the challenges mentioned above.

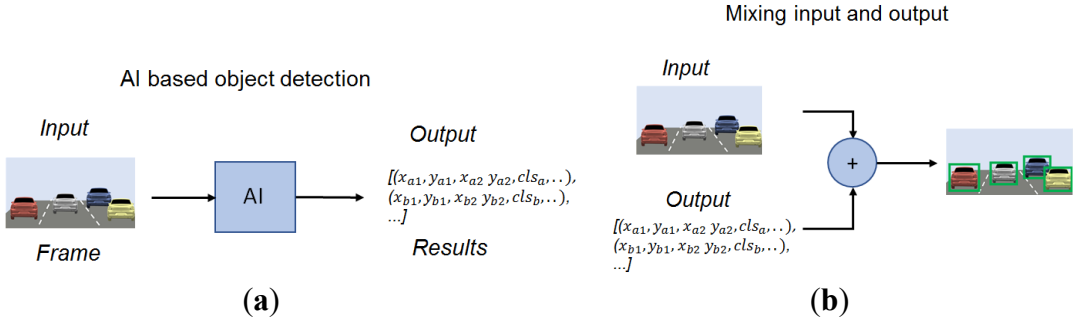
Figure 1 is devoted to the traditional AI-based object detection process and the way of representation of its result. Figure 1(a) represents the traditional AI-based object detection working principle, while Figure 1(b) indicates the way of representation of the result of Figure 1(a).

Figure 1(a) illustrates a basic overview of an AI-based object detection system. The system receives an input, which is a frame from a video or an image; in this example, the input frame shows cars on a road. The input frame is passed into an AI model, usually a neural network designed for object detection, such as YOLO (You Only Look Once), SSD (Single Shot MultiBox Detector), or Faster R-CNN. The AI model processes the input to detect and identify objects within the frame. The AI model outputs a list of detected objects in the frame, with each object represented by a set of coordinates and a class label:

$$(x_{a1}, y_{a1}, x_{a2}, y_{a2}, cls_a, \ldots) \quad (1)$$

Here, x_{a1}, y_{a1} and x_{a2}, y_{a2} denote the coordinates of the bounding box around the detected object, while cls_a represents the class label indicating the type of object (e.g., car, pedestrian, etc.). This process is repeated for each detected object, resulting in multiple sets of coordinates and class labels. Overall, the diagram demonstrates how an AI-based object detection system works by taking an input frame, processing it with AI, and providing results with detected objects and their corresponding details.

Figure 1(b) illustrates the concept of mixing input and output in an AI-based object detection system. The system starts with an input frame, which is an image or a video frame showing a scene, such as cars on a road. The AI model processes this input to detect objects, resulting in an output that contains a list of detected objects. Each object is represented by coordinates of the bounding box ($x_{a1}, y_{a1}, x_{a2}, y_{a2}$) and a class label ($cls$), indicating the type of object (e.g., car, pedestrian). The figure shows a process where the input frame and the detection output are combined visually by overlaying the bounding boxes and class labels from the output onto the original input frame, represented by the "+" symbol. The resulting image on the right side shows the input frame with green bounding boxes around the detected objects, corresponding to the coordinates provided in the output, highlighting where the objects are located and what they are. Overall, the diagram demonstrates how the detected results (output) are visually superimposed on the input frame, allowing a clear and intuitive understanding of where objects have been identified by the AI model.



DOI: 10.65322/j.srd.04182026.26

Figure 1. Traditional AI-based object detection. (a) The traditional AI-based object detection process. (b) The resulting output

3.2. Proposed frame comparison method for real-time object detection

This section introduces a novel frame comparison method for real-time object detection in Intelligent Transportation Systems (ITS). For comparison, we also describe the existing SSIM-based method for real-time object detection. Detailed comparison information about both methods is provided in Section 5.

One of the main advantages of the proposed method compared to the traditional AI-based methods is adaptability to resource-constrained systems. The proposed approach tackles challenges by using algorithms that adapt the frame comparison criteria according to the context and movement within the scene. This adaptability keeps the system responsive and accurate, especially in highway traffic situations.

The following information serves as a brief introductory overview of the proposed method.

When explaining the proposed method at the introductory level (see Figure 2), the following condition can be conditionally accepted (see Equations 2-3):

If

$$\frac{dv_1}{dt} \approx \frac{dv_2}{dt} \approx \frac{dv_3}{dt} \approx \frac{dv_4}{dt} \approx \frac{dv_5}{dt} \tag{2}$$

Then

$$\frac{dS_1}{dt} = c_1; \qquad \frac{dS_2}{dt} = c_2; \qquad \frac{dS_3}{dt} = c_3; \qquad \frac{dS_4}{dt} = c_4; \qquad c_i = const \tag{3}$$

The equations (see equations 2-3) represent a mathematical model for predicting vehicle behavior on a road, assuming equal acceleration rates and constant acceleration for all vehicles. These equations describe the rate of change of distance between the ego vehicle and other vehicles on the road.

Figure 2 presents a group view of cars/vehicles to illustrate the fundamental concept of the proposed method.

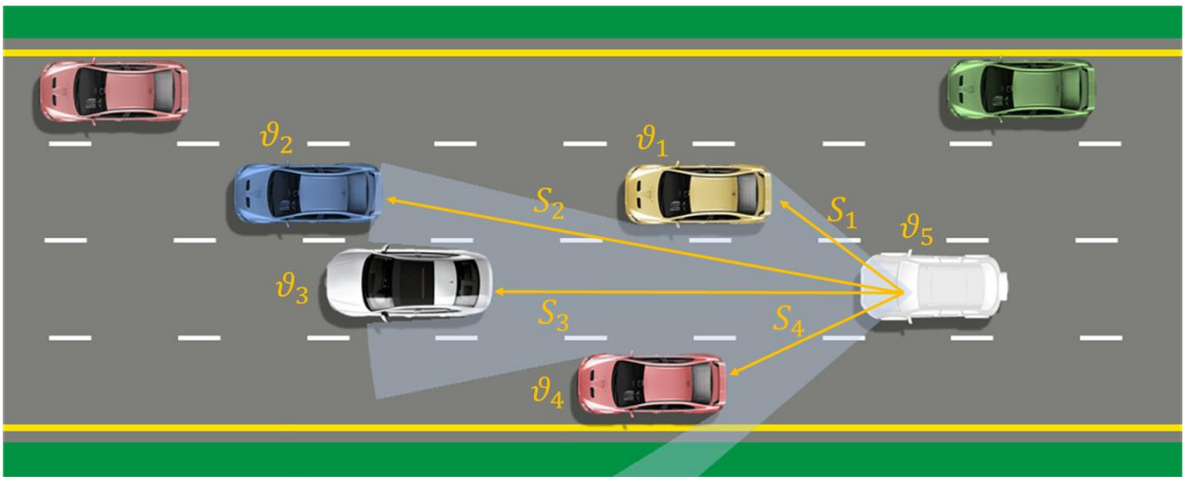


Figure 2. Group view of cars/vehicles from the top for an explanation of the proposed method

Figure 3 is a logical continuation of Figure 2. Figure 3 illustrates the detailed visual tracking of vehicles detected in the forward path of the driving car. As depicted in Figure 3, from Frame 1 (F_1) to Frame 30 (F_{30}), there is almost the same representation between consecutive frames in

DOI: 10.65322/j.srd.04182026.26

tracking cars moving ahead. When estimating the difference between consecutive frames of F_2 and F_1, the difference between these two frames ($\Delta F_{2,1}$) can be determined by matching a threshold (TH). In this case, since $\Delta F_{2,1}$ is smaller than TH, F_2 and F_1 are considered to have almost the same result and can be defined as $R_2 = R_1$. Thus, if we apply this technique to F_{30}, it will give the same result. When there is a variation between consecutive frames in tracking cars moving ahead, the result (R) changes and we can conditionally call it updating/new result. This can be seen in the transition from F_{30} to F_{31}. In this case, $\Delta F_{16,15}$ will be greater than TH, which will be taken as a new result and denoted by R_{16}. Also, there is a updating/new result in the transition from F_{n-1} to F_n. In this case, $\Delta F_{n,n-1}$ will be greater than TH, and it can be denoted by R_n.

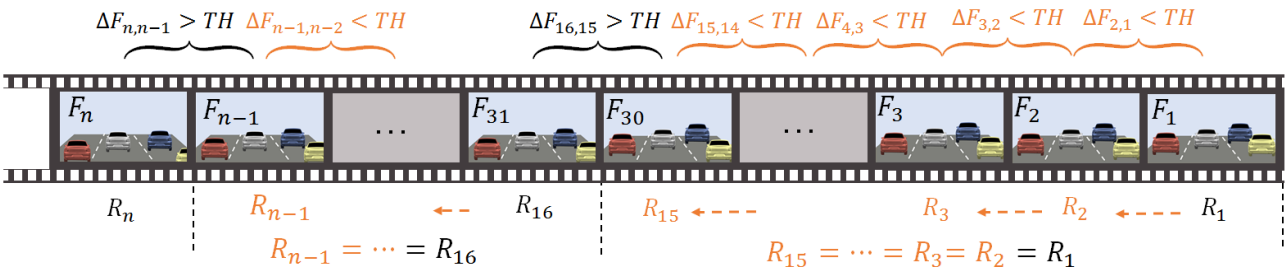


Figure 3. Visual tracking of vehicles detected in the forward path of the driving car

Figure 4 illustrates frame filtering techniques utilized in object detection (OD).

Initially, start, we provide a concise overview of the SSIM-based object detection system using frame filtering, shown in Figure 4(a).

As depicted in Figure 4(a), the OD system presented in this figure uses the structural similarity index measure (SSIM) metric, which is a widely used metric in image processing to measure the similarity between two images. A brief technical explanation of this metric is presented next.

SSIM measures how closely a recovered image or video matches its original [32-34].

SSIM is defined as:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y+c_1)(2\sigma_{xy}+c_2)}{(\mu_x^2+\mu_y^2+c_1)(\sigma_x^2+\sigma_y^2+c_2)} \quad (4)$$

Here: x is frame one, in our case F_n;

y is frame two, in our case F_{n-1};

μ_x the average of x;

μ_x the average of y;

σ_x^2 the average of x;

σ_y^2 the average of y;

$2\sigma_{xy}$ the covariance of x and y;

$c_1 = (k_1L)^2, c_2 = (k_2L)^2$ two variables to stabilize the division with weak denominator;

L = the dynamic range of the pixel-values (typically this is $2^{\#bits\ per\ pixel} - 1$)

$k_1 = 0.01$ and $k_2 = 0.03$ by default

As depicted in Figure 4(b), proposed method based frame filtering OD system proposes a new frame comparison metric. A brief technical explanation of this metric is presented next.

$$Proposed(x,y) = 1 - \frac{\mu_{|x-y|}}{255} \quad (5)$$

Here: x is frame one, in our case F_n

DOI: 10.65322/j.srd.04182026.26

y is frame one, in our case F_{n-1}

$\mu_{|x-y|}$ the average of $|x-y|$

We propose a novel approach for processing and analyzing frames, which are sequential units of information (see Figure 4(b)). Our method involves comparing consecutive frames to identify significant changes and applying a specialized algorithm to process frames exhibiting minimal change.

In the proposed method, the following processes occur:

1. Input: A sequence of data frames.
2. Processing:

- Frame Comparison: A comparison between the current frame (F_n) and the previous frame (F_{n-1}) to determine the degree of change.
- Threshold-Based Filtering: A threshold (TH) is established to filter out frames with insignificant changes.
- Proposed Method Application: For frames with changes below the threshold, a tailored algorithm is applied.
- AI Integration: For frames with changes exceeding the threshold, an AI component is engaged for more complex analysis.

3. Output: The final result, derived from the combination of processed frames and AI-assisted analysis.

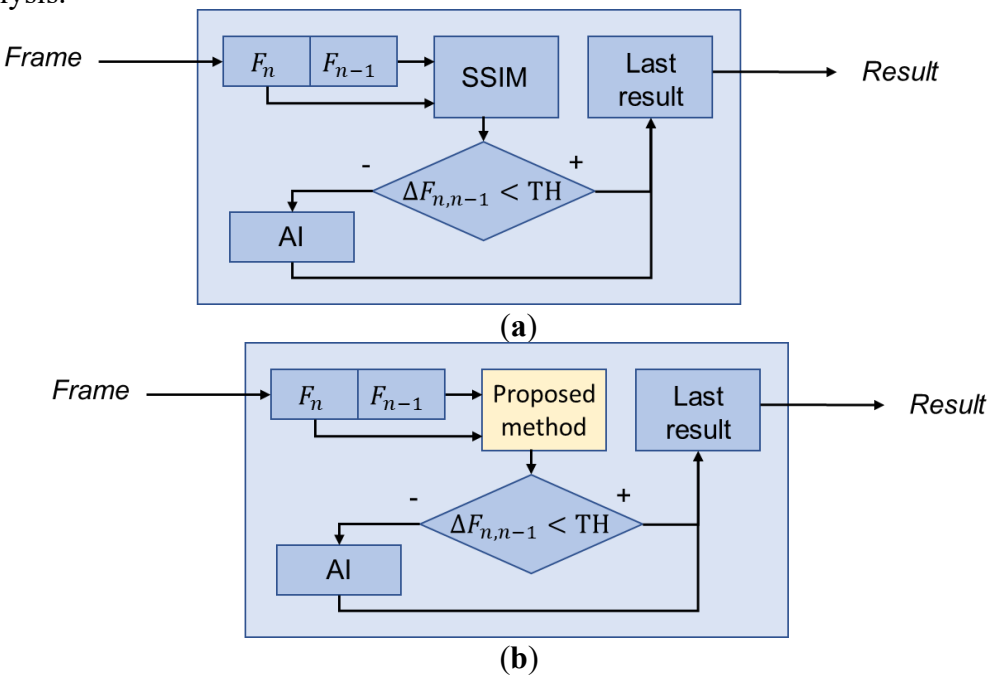


Figure 4. Frame filtering methods for object detection (OD). (a) SSIM based frame filtering OD system. (b) Proposed method-based frame filtering OD system

4. Dataset, Model, and Implementation

This section details the newly developed dataset and the AI architecture employed in this research. Additionally, we present the implementation of our proposed method in real-time object detection.

4.1. Dataset

In this work, we created new dataset to implement the proposed method. Using a single 10-minute highway video, we generated a dataset of 18,000 frames each with resolution of 640x640. The specifications of the developed dataset in this work is shown in Table 1.

Table 1. The specifications of the dataset developed in the study.

DOI: 10.65322/j.srd.04182026.26

№	Name	Dataset
1	Frame resolution	640x640
2	Frame count	18 000
3	Frame rate	30 FPS
4	Video duration	10 min
5	Driving condition	Highway, daytime

4.2. AI model

In this work, the YOLOv8n model was employed for object detection on a dataset comprising 18,000 image sequences, each with a resolution of 640x640 pixels.

YOLOv8n is a state-of-the-art object detection model designed for real-time applications. It is part of the YOLOv8 family, known for its speed and accuracy. YOLOv8n is specifically optimized for smaller devices and resource-constrained environments, making it suitable for embedded systems and mobile applications. Despite its compact size, YOLOv8n maintains impressive performance, achieving high detection rates while maintaining low latency. Due to having these capabilities, it can be used for essential tasks like pedestrian detection, traffic monitoring, and real-time surveillance in ITS.

The specifications of the YOLOv8n is shown in Table 2.

Table 2

YOLOv8n specifications

Parameter	Value
Model Size	Smallest
Inference Speed	Fastest
mAP@0.5	Around 60% (varies depending on dataset)
Intended Use Cases	Low-power devices, real-time applications with limited computational resources

4.3. Implementation

The methods mentioned in Section 3.2 were realized with Python 3.8 using OpenCV, Ultralytics, and Sklearn libraries. The specifications of the server used in the study is detailed in Table 3.

Table 3

The specifications of the server used in the study

CPU	Intel Core i7-8700 @4.6 GHz
GPU	NVIDIA GeForce GTX 1080
Memory (RAM)	64 GB
Operating System	Windows 10 (64 bits)

5. Evaluation

In this section, we conduct a comprehensive benchmarking analysis comparing the proposed frame comparison object detection method with the SSIM-based object detection method.

5.1. Evaluation metrics

The evaluation focuses on three key performance indicators and well-known object detection metric: total processing time (TPT), mean total processing time (mTPT), and skipped frames (SF); and mAP; respectively. We aim to highlight the potential benefits and efficiency gains of the proposed frame comparison object detection method over the SSIM-based object

DOI: 10.65322/j.srd.04182026.26

detection method by comparing these indicators. This analysis offers valuable insights into the method's effectiveness, particularly in managing complex computational tasks where data handling and processing speed are critical.

Mean Average Precision (mAP) is a common metric used to evaluate the performance of object detection models. It measures the accuracy of the model by averaging the precision across different recall levels and then taking the mean across all classes.

Average Precision (AP) is used to denote mAP. Therefore, we explained first AP by equation.

AP is explained by Equation 6:

$$AP = \sum_{n}^{N}(R_n - R_{n-1})P_n \quad (6)$$

Where: P_n is the precision at the n-th threshold.

R_n is the recall at the n-th threshold. N is the number of thresholds.

Mean Average Precision (mAP) is explained by Equation 7:

$$mAP = \frac{1}{C}\sum_{c=1}^{C} AP_c \quad (7)$$

Where: C is the total number of classes. AP_c is the Average Precision for class c.

Figure 5 illustrates the limitations of traditional AI-based object detection method in real-time object detection by analysing the mean total processing time (mTPT) versus frame count (FC) on 640x640 resolution. All frame counts equal 18,000 (see Section 4.1.). For (640x640) resolution, mTPT was 10.86 ms.

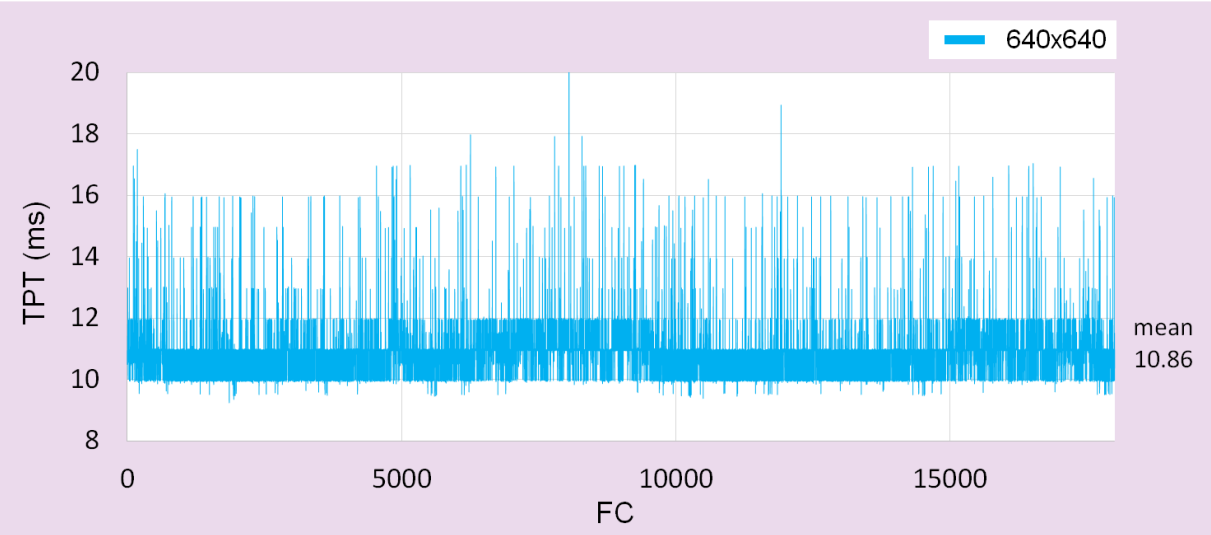


Figure 5. Traditional AI-based object detection: TPT vs. FC

Figure 6 compares the performance of two methods, SSIM (orange) and proposed method (green), using four metrics: Skipped Frames (SF), mean Total Processing Time (mTPT), mean Average Precision at 50% IoU (mAP50), and mean Average Precision across 50-95% IoU (mAP50-95). These metrics are plotted against different threshold values (TH) at a resolution of 640x640 pixels.

In the top-left graph, the percentage of Skipped Frames (SF) is plotted against TH. Skipped frames for SSIM increase sharply, with SF reaching 100% by a threshold of around 40. In contrast, the proposed method maintains a significantly lower percentage of skipped frames until TH

DOI: 10.65322/j.srd.04182026.26

reaches about 40, after which it increases more gradually. This indicates that the proposed method is better at avoiding skipped frames, particularly at lower thresholds.

The top-right graph shows the mean Total Processing Time (mTPT) in milliseconds (ms) against the threshold. SSIM starts with a higher processing time and shows a gradual decline in mTPT as TH increases, eventually stabilizing around 40 ms beyond TH ≈ 40. The proposed method, however, demonstrates a consistently lower processing time, remaining nearly flat across all thresholds. This highlights the proposed method's efficiency in maintaining a low and stable processing time compared to SSIM.

In the bottom-left graph, the mean Average Precision at 50% Intersection over Union (mAP50) is plotted as a percentage. SSIM shows a steady decline in precision as the threshold increases, dropping sharply after TH ≈ 20, with mAP50 falling below 40% by TH ≈ 40. The proposed method performs significantly better, retaining high precision until TH reaches 40. Even after this point, its performance is superior to SSIM, indicating better precision at higher thresholds.

The bottom-right graph focuses on the mean Average Precision at 50-95% IoU (mAP50-95). Similar to mAP50, SSIM sees a decline starting after TH ≈ 20, with performance dropping sharply beyond this point. The proposed method, however, performs better across the entire range of thresholds, maintaining higher mAP50-95 values compared to SSIM, though there is a gradual decline as TH increases.

Overall, the proposed method outperforms SSIM in all key areas: it significantly reduces skipped frames, maintains a consistently lower processing time, and retains higher precision across both mAP50 and mAP50-95 metrics, particularly at higher thresholds. This demonstrates that the proposed method is more efficient and accurate in handling image processing tasks, especially in scenarios with higher threshold values.

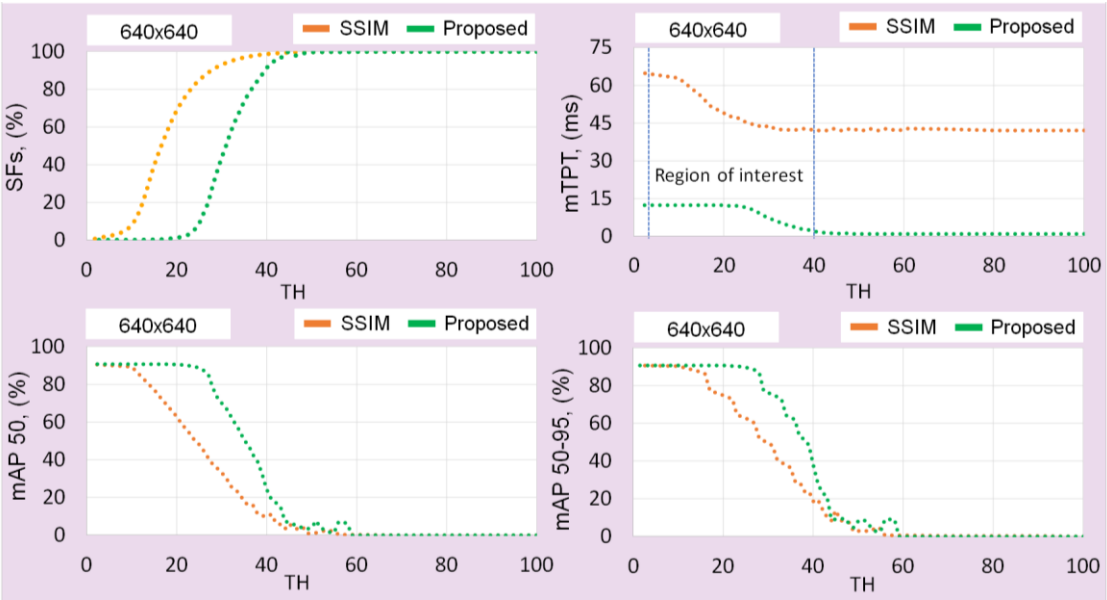


Figure 6. Comparison of mTPT, mAP, and SF in different threshold (TH) values between the SSIM based method and the proposed method

Figure 7 illustrates the comparison of Total Processing Time (TPT) between the SSIM method (orange) and the proposed method (green) over different thresholds (TH = 40, 30, 20, 10) at a resolution of 640x640 pixels. The x-axis represents the Frame Count (FC), while the y-axis displays the TPT in milliseconds (ms). The goal is to highlight the stability and efficiency of each method in processing frames over time as the threshold changes.

DOI: 10.65322/j.srd.04182026.26

At TH = 40 (top-left), SSIM shows significant fluctuation in processing times, with multiple spikes reaching as high as 100 ms and occasional peaks exceeding this value. The average TPT for SSIM remains high, displaying erratic behavior across the entire frame range. In contrast, the proposed method demonstrates much more consistent processing times, staying well below 50 ms throughout. This consistent performance underlines the efficiency of the proposed method, which avoids the sharp increases in processing time that SSIM experiences at high thresholds.

In the TH = 30 case (top-right), SSIM continues to exhibit significant variability in TPT, although the frequency of extreme spikes above 100 ms decreases slightly compared to TH = 40. The average TPT for SSIM remains around 50-100 ms, indicating that it still suffers from processing inefficiencies. The proposed method, however, maintains its low and stable processing time, once again staying below 50 ms with minimal fluctuations. This consistency showcases the resilience of the proposed method in handling frame processing even as the threshold decreases.

At TH = 20 (bottom-left), the SSIM method's TPT still fluctuates significantly, but with fewer spikes exceeding 100 ms. The average TPT for SSIM, however, remains unstable and varies between 50-100 ms. In contrast, the proposed method continues to show superior performance, maintaining low TPT values under 50 ms across the frame count range. The consistency of the proposed method's performance becomes more pronounced as the threshold decreases, highlighting its ability to handle increasing computational demands more efficiently than SSIM.

Finally, at TH = 10 (bottom-right), while SSIM shows a reduction in the number of high TPT spikes, the overall processing time remains high, fluctuating between 50 and 100 ms. The variability in SSIM's performance persists, though the peaks are less extreme than at higher thresholds. The proposed method, however, performs even more efficiently at this lower threshold, with TPT consistently below 30 ms. The stability of the proposed method, even at the lowest threshold, reinforces its advantage in providing faster and more reliable processing times compared to SSIM.

Across all thresholds, the proposed method consistently outperforms SSIM in terms of TPT, demonstrating lower and more stable processing times. As thresholds increase, SSIM struggles with significant fluctuations and spikes in TPT, especially at higher thresholds like TH = 40 and 30. Meanwhile, the proposed method remains unaffected by such variations, consistently delivering faster and more efficient frame processing. This makes the proposed method superior for applications requiring high processing speed and stability, regardless of threshold values.

DOI: 10.65322/j.srd.04182026.26

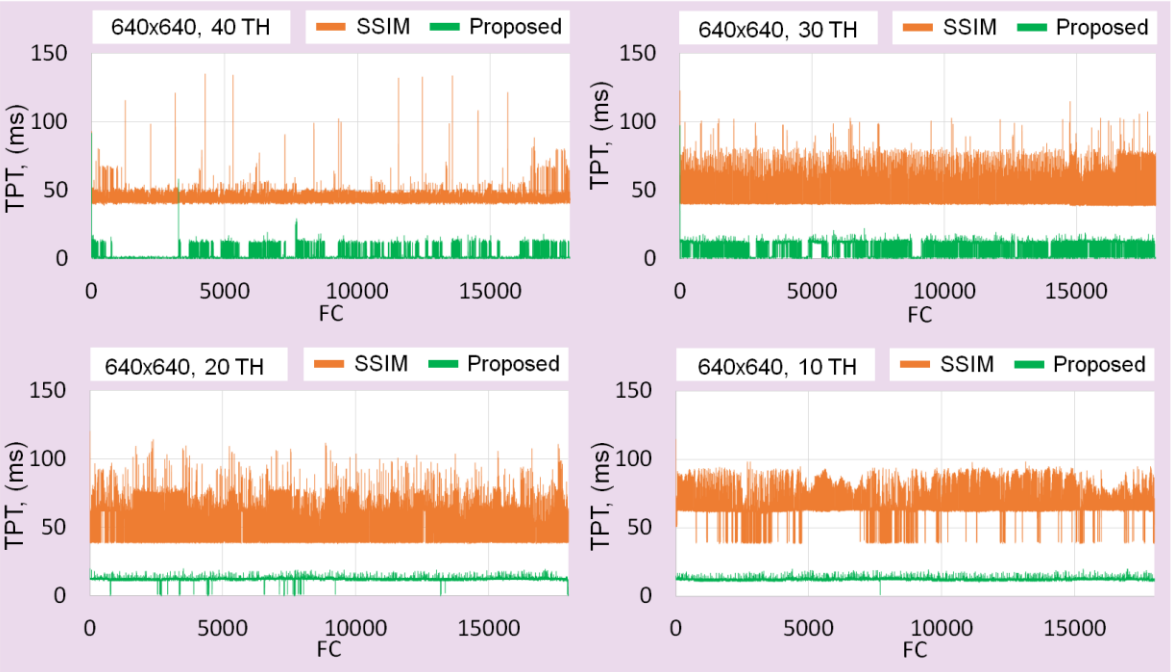


Figure 7. Comparison of TPT in different TH values between the SSIM based method and the proposed method

Figure 8 provides a detailed comparison between the SSIM (orange) and the Pro-posed (green) methods in terms of mean Total Processing Time (mTPT) and mean Average Precision (mAP), at a resolution of 640x640 pixels. The two subplots represent the relationship between mTPT (in milliseconds) and two key precision metrics: mAP50 and mAP50-95. These metrics measure precision at different Intersection over Union (IoU) thresholds, with mAP50 corresponding to an IoU threshold of 0.5 and mAP50-95 representing the average precision across IoU thresholds from 0.5 to 0.95.

In the left subplot, the mAP50 is plotted against mTPT. The SSIM method shows a gradual rise in mAP50 from 0 ms to around 45 ms, reaching a maximum mAP50 of about 80%. This suggests that SSIM requires considerable processing time to achieve moderate precision. On the other hand, the proposed method demonstrates a more efficient performance, achieving an mAP50 of 90% in less than 15 ms. This difference clearly indicates that the proposed method provides higher precision with a much lower computational overhead, making it a more effective option for applications requiring fast and accurate image processing.

The right subplot presents the relationship between mTPT and the mAP50-95 metric, which reflects precision across a wider range of IoU thresholds. Again, SSIM shows an increase in mAP50-95 as the processing time increases, reaching just over 60% mAP50-95 after nearly 45 ms. In contrast, the proposed method achieves significantly better results, reaching 85% mAP50-95 with a processing time of less than 15 ms. This clearly demonstrates that the proposed method not only reduces processing time but also improves precision when dealing with varying IoU thresholds.

Across both precision metrics, the proposed method significantly outperforms SSIM. It consistently delivers higher precision values while maintaining processing times that are approximately three times faster. These results underscore the superiority of the proposed method, especially in environments where both accuracy and computational efficiency are critical.

DOI: 10.65322/j.srd.04182026.26

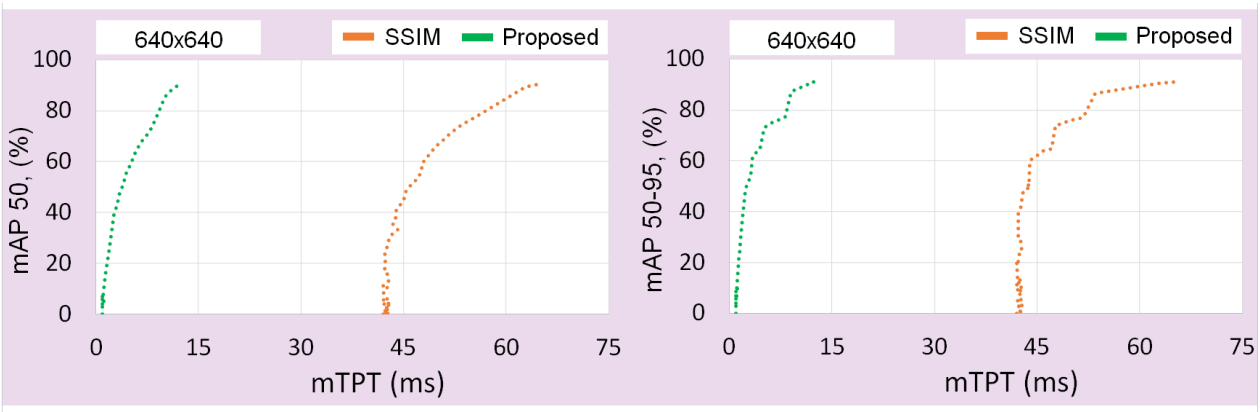


Figure 8. Comparison of mTPT (in ms) and mAP values (in %) between the SSIM-based and pro-posed methods

5.2. Validation of experimental results

This section provides validation of experimental results of the proposed method. To validate the results obtained in this study, root mean square error (RMSE) was used to evaluate the error between the proposed method and the mathematically expected results. RMSE is a variation of MSE that measures the square root of the average squared difference between corresponding pixels in two images. The RMSE is defined as (see Equation (8)):

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(I_1 - I_2)^2}{n}} \tag{8}$$

Where, n is the number of pixels, and I_1 and I_2 are the corresponding pixel values of the two images [32-34].

Bifurcation for 25000 different values of k in range 0.1-0.6.

The following equation (see Equation (9)) was used to express mAP50.

$$mAP50 = b - \frac{L}{1+e^{(-k\cdot(TH-x_0))}} \tag{9}$$

Where:

L is responsible for scaling the output range from [0,1] to [0, L], max(mAP) – 91.0854

b adds bias to the output and changes its range from [0, L] to [b, L + b] - 1.2372

k is responsible for scaling the input, 0.2409

x_0 is the point in the middle of the Sigmoid, median(mAP) – 35.6795

RMSE=2.4029

Figure 9 compares the mathematically expected results with the results from the proposed method across various threshold (TH) values.

Figure 9(a) shows the performance of two object detection methods, comparing the mathematically expected results (orange solid line) with the results from the proposed method (green dotted line) across varying threshold (TH) values. The x-axis represents the threshold (TH), which likely controls the overlap criteria for correct detections, while the y-axis shows the mean Average Precision (mAP50), a common evaluation metric in object detection, expressed as a percentage. The experiment is based on 640x640 pixel images. At low thresholds (TH < 30), both the mathematical model and the proposed method maintain a high accuracy, with mAP50 values above 80%. This indicates that both methods are performing well under lenient overlap conditions, capturing most object detections effectively.

DOI: 10.65322/j.srd.04182026.26

As the threshold increases, particularly in the range of 30 < TH < 50, a noticeable drop in mAP50 occurs for both methods. This decline suggests that the overlap criteria become stricter, making it harder to meet the detection threshold, thus reducing the accuracy of the predictions. The proposed method follows the mathematical model closely but exhibits slight fluctuations, especially around the 30–40 threshold range, where the green dotted line shows some deviation. These fluctuations indicate that while the proposed method is largely accurate, it may handle certain edge cases differently than the theoretical expectation. Beyond a threshold of 50, both methods show a significant drop in performance, with the mAP50 approaching zero, signifying that very few detections meet the stricter overlap criteria at higher thresholds.

The proposed method performs nearly as expected based on the mathematical model, with only minor deviations observed at specific thresholds. The close alignment between the two methods indicates the robustness of the proposed approach.

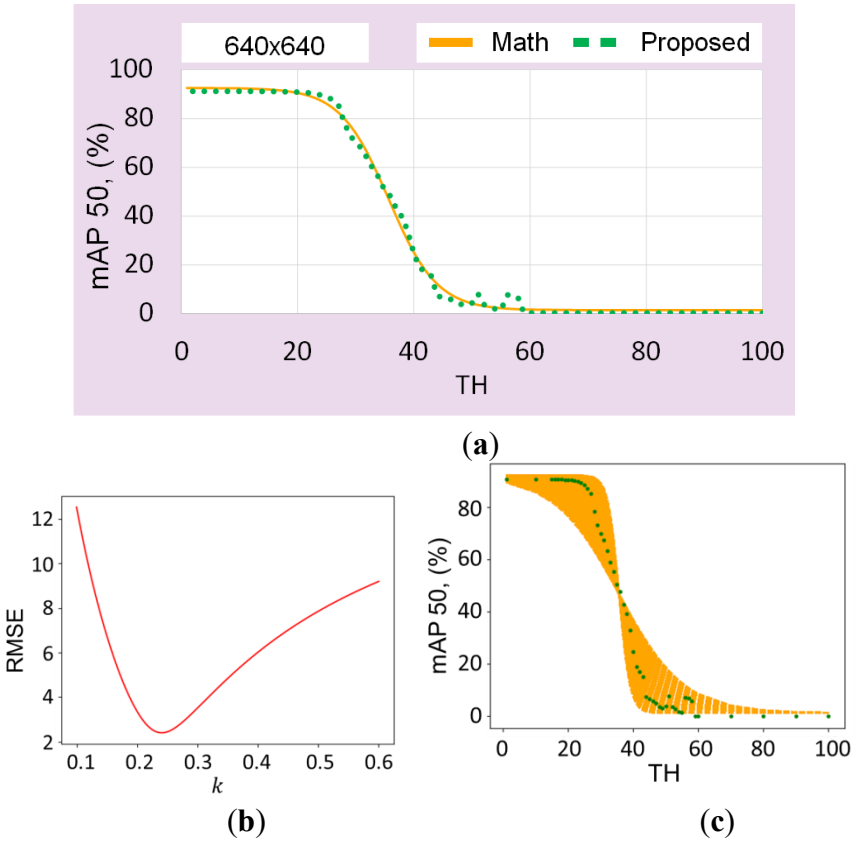
Figure 9(b) illustrates a comparison between the performance of the proposed method (green dots) and the mathematically expected results based on a sigmoid function (orange shaded area) across various thresholds (TH). The x-axis represents the threshold, while the y-axis shows the mean Average Precision (mAP50) in percentage, a common metric for evaluating object detection accuracy. At low thresholds (TH < 30), the proposed method achieves high mAP50 values above 80%, meaning that it successfully detects objects when the overlap criterion between predicted and ground truth boxes is lenient. The green dots closely align with the sigmoid-based expected results, indicating that the proposed method performs as predicted by the mathematical model in these cases, with minimal deviations.

As the threshold increases to the mid-range (30 < TH < 50), both the mathematically expected results and the proposed method show a sharp decline in mAP50 values. The orange region narrows as the overlap criterion becomes stricter, and the green dots begin to deviate slightly, particularly around TH values of 40. These deviations suggest that while the proposed method still performs within the expected range, it struggles more as the threshold increases. The spread in the orange area reflects the expected variability in the sigmoid model, and the green dots indicate that the proposed method generally conforms to these predictions but with slight fluctuations in this more challenging threshold range.

At higher thresholds (TH > 50), both the mathematically expected results and the proposed method converge toward a near-zero mAP50, indicating that neither method can meet the stricter overlap requirements. The green dots remain largely within the lower part of the orange shaded region, showing that the proposed method continues to follow the sigmoid trend even as detection accuracy drops significantly. Overall, the graph demonstrates that the proposed method performs well in line with the sigmoid-based mathematical expectations, with only minor deviations observed as the threshold becomes more restrictive.

Figure 9(c) displays the Root Mean Square Error (RMSE) curve, with the x-axis representing the parameter k and the y-axis showing the RMSE values. The RMSE of 2.4029 reflects the error between the results obtained from the proposed method and the mathematically expected outcomes. The curve indicates that around k=0.2, the RMSE reaches its lowest point, signifying that this is the optimal value where the proposed method closely aligns with the theoretical results, yielding the smallest error. As k increases beyond this point, the RMSE rises, suggesting a growing deviation between the proposed method's predictions and the mathematical expectations, which implies a loss of accuracy at higher k values. This pattern highlights the importance of selecting an optimal k value to minimize errors and ensure alignment between the proposed and expected results.

DOI: 10.65322/j.srd.04182026.26



(**a**) Comparison of performance analysis of the mathematically expected results versus the proposed method across different threshold (TH) values. (b) Comparison between the performance of the proposed method and the mathematically expected results based on a sigmoid function across various thresholds (TH) and mAP50. (**c**) Difference (error) between the results obtained from the proposed method and the mathematically expected outcomes

Figure 9. Comparison of the mathematically expected results with the results obtained using the proposed method across various threshold (TH) values

Bifurcation for 25000 different values of k in range 0.1-0.6.

The following equation (see Equation (10)) was used to express mTPT.

$$mTPT = b + \frac{L}{1+e^{(-k\cdot(TH-x_0))}} \tag{10}$$

Where L is responsible for scaling the output range from [0,1] to [0, L], max(mTPT) – -11.6577

b adds bias to the output and changes its range from [0, L] to [b, $L + b$] – 12.6356

k is responsible for scaling the input, 0.29

x_0 is the point in the middle of the Sigmoid, median(mTPT) – 31.0339

RMSE=0.1888.

Figure 10 compares the mathematically expected results with the results from the proposed method across various threshold (TH) values with different mTPT.

Figure 10(a) illustrates the mean Total Processing Time (mTPT) in milliseconds (ms) as a function of the threshold (TH), comparing the results from the mathematically expected model (orange solid line) and the experimental results based on the proposed method (green dotted line). The x-axis represents the threshold values (TH), while the y-axis represents the mTPT. The experiments were conducted using images with a resolution of 640x640 pixels. At lower thresholds (TH < 30), both the mathematical model and the proposed method show similar mTPT values of

DOI: 10.65322/j.srd.04182026.26

around 12 ms, indicating that the processing times are almost identical in this region. This suggests that under less strict detection conditions, the proposed method performs as efficiently as the mathematically modeled expectation, maintaining a high level of agreement.

As the threshold increases into the range of 30 to 50, there is a noticeable drop in mTPT for both the mathematical model and the experimental results. By TH = 50, the mTPT falls to around 5 ms, reflecting the reduced computational load as higher thresholds limit the number of processed detections. The alignment between the proposed method and the mathematical model remains strong, with both exhibiting similar reductions in processing time as the threshold increases. This reduction is significant because it indicates that the proposed method scales well in terms of efficiency as stricter criteria are applied.

Beyond TH = 50, both the proposed method and the mathematical model level off at mTPT values near 3 ms, indicating that the processing time stabilizes even as the threshold continues to increase. This suggests that at high threshold values, both methods maintain their efficiency without any significant difference in performance. Overall, the proposed method closely mirrors the expected mathematical behavior, with minimal deviations throughout the threshold range, confirming the method's ability to optimize processing time under varying detection conditions.

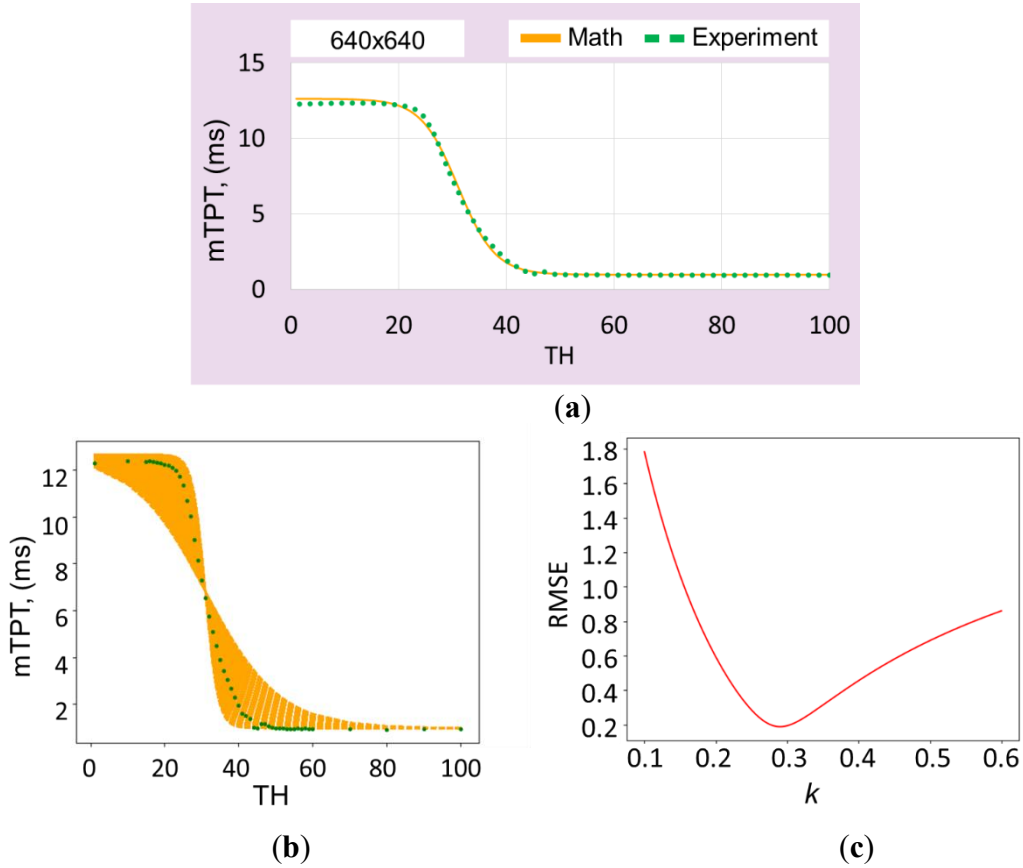
Figure 10(b) compares the performance of the proposed method (green dots) against mathematically expected results based on a sigmoid function (orange shaded area). At lower threshold values (TH < 30), both the proposed method and the mathematical model produce similar processing times, with mTPT values around 11 ms. This close alignment suggests that when the detection threshold is less strict, the proposed method performs efficiently, closely matching the theoretical expectations. As the threshold increases, both methods experience a significant drop in processing time, with mTPT decreasing sharply to around 4 ms by the time TH reaches 40, reflecting the reduced computational load as the threshold becomes more stringent.

For higher thresholds (TH > 40), the mTPT values for both the proposed method and the sigmoid-based model level off around 1 ms. The green dots representing the proposed method remain within the orange-shaded region, indicating that the method performs consistently with the expected mathematical behavior across the threshold range. While minor fluctuations between the experimental and expected results occur, particularly around TH = 40, the overall alignment remains strong. This demonstrates that the proposed method scales well in terms of efficiency as the detection threshold increases, maintaining close adherence to the theoretical model's performance throughout.

Figure 10(c) shows the Root Mean Square Error (RMSE) as a function of the parameter k, where the x-axis represents the values of , and the y-axis displays the RMSE values. The RMSE indicates the error between the results obtained from the proposed method and the mathematically expected outcomes. In this case, the minimum RMSE achieved is 0.1888, which represents the smallest difference between the proposed method's results and the expected results.

The curve follows a U-shape, showing that as k increases from around 0.1 to 0.3, the RMSE decreases, reaching its lowest value at k = 0.3. This suggests that around k=0.3, the proposed method aligns most closely with the mathematically expected results, minimizing error. Beyond this point, as k increases further, the RMSE begins to rise again, indicating that the performance of the proposed method starts to deviate more from the expected results. This U-shaped curve emphasizes the optimal k value of approximately 0.3, where the proposed method achieves its highest accuracy relative to the mathematical model.

DOI: 10.65322/j.srd.04182026.26



(**a**) (**b**) (**c**)

Figure 10. Comparison of the mathematically expected results with the results from the proposed method at various threshold (TH) values and different mTPT. (a) Comparison of performance analysis of the mathematically expected results versus the proposed method across different threshold (TH) values and mTPT. (b) Comparison between the performance of the proposed method and the mathematically expected results based on a sigmoid function across various thresholds (TH) and mAP50. (c) Difference (error) between the results obtained from the proposed method and the mathematically expected outcomes.

Bifurcation for 25000 different values of b in range 150-175.

The following equation based on logarithmic function (see Equation (11)) was used to express mAP50.

$$mAP50 = a \cdot \log(b \cdot mTPT) + c \tag{11}$$

The best fit values of all parameters:

a = 34.4740, b = 163.6600, c= -171.7773, RMSE = 1.9008.

Figure 11(a) compares the mean Average Precision at 50% overlap (mAP50), expressed as a percentage, with the mean Total Processing Time (mTPT) in milliseconds (ms). The orange solid line represents the mathematically expected results, while the green dotted line reflects the performance of the proposed method. Both methods show a strong correlation between increased processing time and improved detection accuracy. At lower mTPT values (below 5 ms), both curves rise sharply, with mAP50 increasing from 0% to approximately 60%. The proposed method aligns closely with the mathematical expectations in this region, demonstrating that it effectively detects objects with minimal deviation from theoretical predictions when processing time is limited.

As the mTPT increases beyond 5 ms, the mAP50 continues to improve for both methods, reaching values close to 90% as the processing time approaches 10 ms. While the proposed method

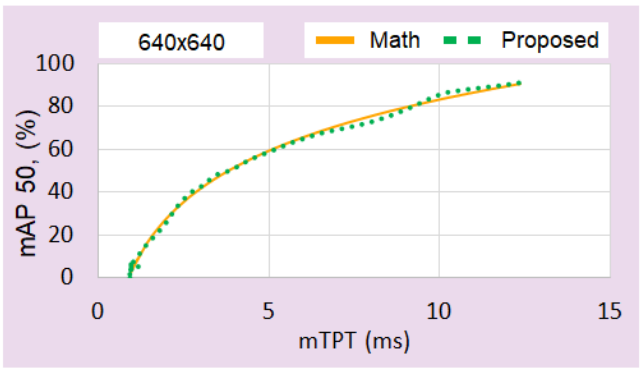
DOI: 10.65322/j.srd.04182026.26

shows slight fluctuations, it remains closely aligned with the mathematically expected results. This indicates that the proposed method performs almost as well as the mathematical model, with only minor differences observed in the higher processing time range. Overall, the graph demonstrates that the proposed method consistently achieves high detection accuracy across varying processing times, closely matching the theoretical expectations.

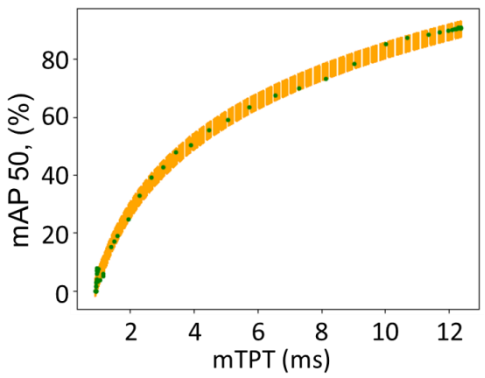
Figure 11(b) shows a comparison of mAP50 in percentage against mTPT in milliseconds (ms), with the mathematically expected results modeled by a logarithmic function (orange shaded area) and the proposed method results represented by green dots. Both curves follow a logarithmic growth pattern, where mAP50 increases as processing time increases. For shorter processing times (mTPT < 4 ms), both methods show relatively low accuracy, around 10%, but improve quickly as mTPT increases. In this region, the proposed method aligns closely with the expected results, demonstrating that it achieves comparable performance to the mathematical model with only slight deviations.

As the mTPT increases beyond 4 ms and approaches 12 ms, mAP50 continues to rise, reaching values near 90%. Throughout this range, the proposed method remains within the orange shaded region, indicating that its performance is consistent with the logarithmic growth trend predicted by the mathematical model. Minor fluctuations between the proposed method and the expected results are observed but do not significantly affect the overall alignment.

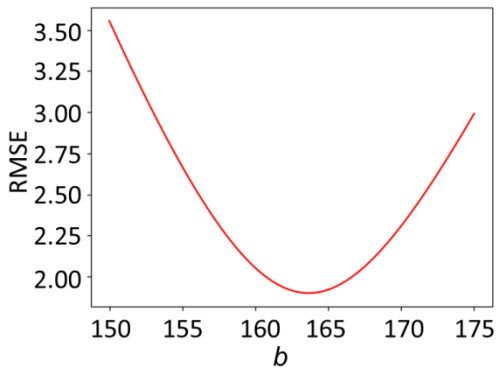
Figure 11(c) depicts the Root Mean Square Error (RMSE) as a function of the parameter b, with the x-axis representing values of b and the y-axis showing the RMSE.



(a)



(b)



(c)

(a) Comparison of performance analysis of the mathematically expected results versus the proposed method across different mAP50 levels and different mTPT. (b) Comparison between the performance of the proposed method and the mathematically expected results based on a logarithmic function across various mAP50 levels and different mTPT. (c) Difference (error)

DOI: 10.65322/j.srd.04182026.26

between the results obtained from the proposed method and the mathematically expected outcomes

Figure 11. Comparison of the mathematically expected results with the results from the proposed method at various mAP50 levels and different mTPT

The RMSE reflects the error between the results obtained from the proposed method and the mathematically expected outcomes, with a minimum value of 1.9008. The curve follows a U-shape, where the RMSE starts relatively high at around 3.5 for lower values of b (approximately 150), then steadily decreases as b increases. The error reaches its lowest point at b=162, indicating that the proposed method closely matches the mathematically expected results at this value.

As b increases beyond 165, the RMSE begins to rise again, signifying a growing deviation between the proposed method's results and the expected outcomes. This pattern highlights that the parameter b has an optimal range near 162, where the error is minimized, and the proposed method is most accurate. Deviations from this optimal range, either higher or lower, lead to an increase in RMSE, reflecting the method's sensitivity to changes in b. The graph underscores the importance of tuning b within this specific range to achieve the best alignment with theoretical expectations.

Mathematical representation of frame repeating and dependence of its parameters on TH.

The following equation (see Equation (12)) was used mathematical representation of frame repeating and dependence of its parameters on TH.

$$p_d = b + \frac{L}{1+e^{(-k\cdot(fr-x_0))}} \tag{12}$$

Here: p_d - probability density of repeating frames;

fr – frame repeating;

b, L, k, x_0 – parameters.

Figure 12 presents six histograms, each illustrating the Probability Density (PD) of frame repetition as a function of Frame Count (FC) for different threshold (TH) values, ranging from TH = 30 to TH = 40. Each histogram compares the experimental results from the proposed method (green bars) with the mathematically expected results (black line). The x-axis represents the frame count, indicating the number of similar sequential frames, while the y-axis displays the probability density (PD). These visualizations offer insight into how the proposed method aligns with theoretical predictions across various thresholds.

At lower thresholds (TH = 30), the experimental data closely follows the mathematically expected curve, with the majority of the probability density concentrated at lower frame counts (FC < 5). This suggests that both the proposed method and the mathematical model predict frequent occurrences of shorter sequences of repeated frames. As the threshold increases to TH = 32 and TH = 34, the alignment between the experimental and expected results remains strong, though slight deviations begin to appear, particularly in the tails of the distributions. Here, the experimental results tend to show slightly higher probabilities for larger frame counts (FC > 5) compared to the theoretical model.

At higher threshold values (TH = 36 to TH = 40), the experimental data continues to adhere to the general shape of the expected curve, but small discrepancies become more noticeable. The proposed method demonstrates a slightly broader distribution, with a higher probability density for larger frame counts than the mathematical model predicts. Despite these differences, both the experimental and mathematical models maintain the same overall trend, with frame repetition more likely to occur in smaller clusters.

DOI: 10.65322/j.srd.04182026.26

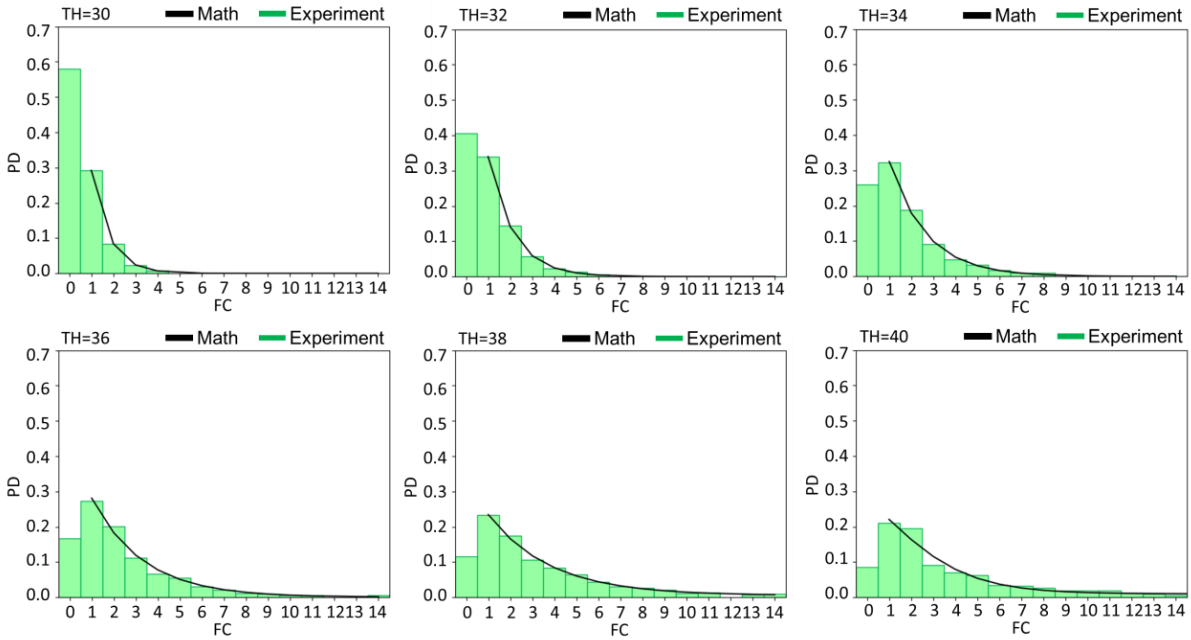


Figure 12. Comparison of the mathematically expected results with the results from the proposed method at various TH, FC and PD values

Figure 13 illustrates the relationship between TH and RMSE. TH represents a threshold value, while RMSE measures the overall accuracy or prediction error of a model. The graph shows that the RMSE values generally decrease as TH increases from 27 to around 35, suggesting that the model's performance improves within this range. However, further increasing the threshold beyond 35 might lead to degraded performance.

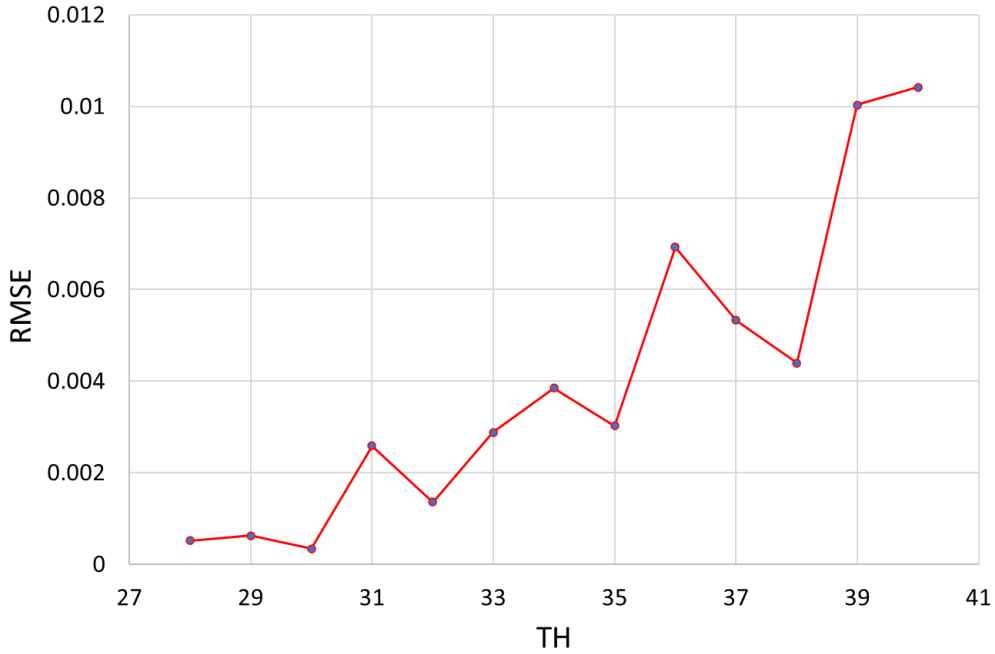


Figure 13. The relationship between TH and RMSE levels

6. Conclusions

DOI: 10.65322/j.srd.04182026.26

In this paper, we introduced an efficient object detection method using a novel frame comparison approach, tailored for Intelligent Transportation Systems (ITS). The method effectively balances computational efficiency and detection accuracy by leveraging frame comparison techniques, making it well-suited for real-time applications such as traffic monitoring and vehicle identification. Across various threshold values, the method demonstrated strong alignment with mathematically expected results, showing reliable performance in maintaining both processing speed and accuracy. However, slight discrepancies were observed, particularly in the probability density of larger frame counts and higher thresholds, where the method overestimated the occurrence of repeated frames. Additionally, as thresholds increased, the rise in RMSE indicated a decline in accuracy under more stringent conditions, which could affect the system's robustness in dynamic ITS environments. Future work will focus on improving the method's performance at higher thresholds to reduce RMSE and enhance accuracy in more demanding scenarios. Exploring adaptive thresholding mechanisms will be a key area to optimize the method's sensitivity to different traffic conditions, improving its flexibility for various ITS applications.

References

1. E. Dilek and M. Dener, "Computer Vision Applications in Intelligent Transportation Systems: A Survey," Sensors, vol. 23, no. 6, 2023, doi: 10.3390/s23062938.

2. Y. Li, L. Chu, Y. Zhang, C. Guo, Z. Fu, and J. Gao, "Intelligent transportation video tracking technology based on computer and image processing technology," J. Intell. Fuzzy Syst., vol. 37, no. 3, pp. 3347–3356, 2019, doi: 10.3233/JIFS-179137.

3. M. Hasanujjaman, M. Z. Chowdhury, and Y. M. Jang, "Sensor Fusion in Autonomous Vehicle with Traffic Surveillance Camera System: Detection, Localization, and AI Networking," Sensors, vol. 23, no. 6, 2023, doi: 10.3390/s23063335.

4. J. Cao et al., "Pedestrian detection algorithm for intelligent vehicles in complex scenarios," Sensors (Switzerland), vol. 20, no. 13, pp. 1–19, 2020, doi: 10.3390/s20133646.

5. L. C. Reveles-Gómez et al., "Detection of Pedestrians in Reverse Camera Using Multimodal Convolutional Neural Networks," Sensors, vol. 23, no. 17, pp. 1–21, 2023, doi: 10.3390/s23177559.

6. M. Sukkar, M. Shukla, D. Kumar, V. C. Gerogiannis, A. Kanavos, and B. Acharya, "Enhancing Pedestrian Tracking in Autonomous Vehicles by Using Advanced Deep Learning Techniques," Inf., vol. 15, no. 2, pp. 1–16, 2024, doi: 10.3390/info15020104.

7. M. Najm and Y. H. Ali, "Automatic vehicles detection, classification and counting techniques / survey," Iraqi J. Sci., vol. 61, no. 7, pp. 1811–1822, 2020, doi: 10.24996/ijs.2020.61.7.30.

8. A. S. Alghamdi, T. Imran, K. T. Mursi, A. Ejaz, M. Kamran, and A. Alamri, "A Vehicle Classification System for Intelligent Transport System using Machine Learning in Constrained Environment," Int. J. Adv. Comput. Sci. Appl., vol. 14, no. 7, pp. 420–432, 2023, doi: 10.14569/IJACSA.2023.0140746.

9. M. Sankaranarayanan, C. Mala, and S. Mathew, "Efficient vehicle detection for traffic video-based intelligent transportation systems applications using recurrent architecture," Multimed. Tools Appl., vol. 82, no. 25, pp. 39015–39033, 2023, doi: 10.1007/s11042-023-14812-4.

10. S. Kumar, M. Jailia, S. Varshney, N. Pathak, S. Urooj, and N. A. Elmunim, "Robust Vehicle Detection Based on Improved You Look Only Once," Comput. Mater. Contin., vol. 74, no. 2, pp. 3561–3577, 2023, doi: 10.32604/cmc.2023.029999.

DOI: 10.65322/j.srd.04182026.26

11. Y. Song et al., “MEB-YOLO: An Efficient Vehicle Detection Method in Complex Traffic Road Scenes,” Comput. Mater. Contin., vol. 75, no. 3, pp. 5761–5784, 2023, doi: 10.32604/cmc.2023.038910.

12. A. R. Youssef, A. A. Ali, and F. R. Sayed, “Real-time Egyptian License Plate Detection and Recognition using YOLO,” Int. J. Adv. Comput. Sci. Appl., vol. 13, no. 7, pp. 853–858, 2022, doi: 10.14569/IJACSA.2022.0130799.

13. S. Tas, O. Sari, Y. Dalveren, S. Pazar, A. Kara, and M. Derawi, “Deep Learning-Based Vehicle Classification for Low Quality Images,” Sensors, vol. 22, no. 13, pp. 1–12, 2022, doi: 10.3390/s22134740.

14. N. L. Yaacob, A. A. Alkahtani, F. M. Noman, A. W. M. Zuhdi, and D. Habeeb, “License plate recognition for campus auto-gate system,” Indones. J. Electr. Eng. Comput. Sci., vol. 21, no. 1, pp. 128–136, 2021, doi: 10.11591/ijeecs.v21.i1.pp128-136.

15. Y. Lyu, I. Schiopu, B. Cornelis, and A. Munteanu, “Framework for Vehicle Make and Model Recognition—A New Large-Scale Dataset and an Efficient Two-Branch–Two-Stage Deep Learning Architecture,” Sensors, vol. 22, no. 21, 2022, doi: 10.3390/s22218439.

16. Z. Mahmood, K. Khan, U. Khan, S. H. Adil, S. S. A. Ali, and M. Shahzad, “Towards Automatic License Plate Detection,” Sensors, vol. 22, no. 3, pp. 1–19, 2022, doi: 10.3390/s22031245.

17. S. Turgay, O. Torkul, R. Model, and P. O. Rate, “Cloud-Based Intelligent Transportation System: Reference Model,” Adv. Comput. Signals Syst., vol. 6, no. 7, pp. 1–9, 2022, doi: 10.23977/acss.2022.060701.

18. M. Pagale, R. Purohit, P. Dhade, A. Thakare, S. Gudadhe, and P. Narkhede, “Insights of Deep Convolutional Neural Network for Traffic Sign Detection in Autonomous Vehicle,” Proc. 2nd Int. Conf. Appl. Artif. Intell. Comput. ICAAIC 2023, no. Icaaic, pp. 176–181, 2023, doi: 10.1109/ICAAIC56838.2023.10141095.

19. S. Li, S. Wang, and P. Wang, “A Small Object Detection Algorithm for Traffic Signs Based on Improved YOLOv7,” Sensors, vol. 23, no. 16, 2023, doi: 10.3390/s23167145.

20. E. Guney and C. Bayilmis, “An Implementation of Traffic Signs and Road Objects Detection Using Faster R-CNN,” Sak. Univ. J. Comput. Inf. Sci., vol. 5, no. 2, pp. 216–224, 2022, doi: 10.1109/ACCESS.2022.3198954.

21. A. Marcella, S. Lee, and B. Seo, “Speeding-up Multiple Object Tracking by Frame Skipping,” in Korea Computer Congress 2018, 2018, no. July, pp. 864–866.

22. P. Shiralagi, R. Bhandary, B. Rajeshwari, and R. Bajarangbali, “An Optimization zation to YOLOv3 YOLOv3-tiny For Real-Time Real Detection of Small , Fast Moving Objects,” Int. J. Res. Sci. Innov., vol. 6, no. X12, pp. 41–46, 2019.

23. J. Dong, J. Yuan, L. Li, X. Zhong, and W. Liu, “Optimizing queries over video via lightweight keypoint-based object detection,” in ICMR 2020 - Proceedings of the 2020 International Conference on Multimedia Retrieval, 2020, vol. 2020, pp. 548–554. doi: 10.1145/3372278.3390714.

24. J. W. Park, J. Kim, and H. J. Lee, “Fast Object Detection Using a Frame Skip Method,” in 2020 IEEE International Conference on Consumer Electronics - Asia, ICCE-Asia 2020, 2020, pp. 30–31. doi: 10.1109/ICCE-Asia49877.2020.9276991.

25. U. Ahmed, J. C. W. Lin, and G. Srivastava, “Unmanned Aerial Multi-Object Dynamic Frame Detection and Skipping Using Deep Learning on the Internet of Drones,” IEEE Internet Things Mag., vol. 4, no. 4, pp. 36–39, 2021, doi: 10.1109/IOTM.001.2100088.

26. J. Lee and K. il Hwang, “YOLO with adaptive frame control for real-time object detection applications,” Multimed. Tools Appl., vol. 81, no. 25, pp. 36375–36396, 2022, doi: 10.1007/s11042-021-11480-0.

DOI: 10.65322/j.srd.04182026.26

27. M. Henning, M. Buchholz, and K. Dietmayer, “The Impact of Frame-Dropping on Performance and Energy Consumption for Multi-Object Tracking,” in 2023 IEEE Intelligent Vehicles Symposium (IV), 2023, pp. 1–6. doi: 10.1109/IV55152.2023.10186653.

28. X. Liu, F. K. Nejadasl, J. C. Van Gemert, O. Booij, and S. L. Pintea, “Objects do not disappear: Video object detection by single-frame object location anticipation,” in 2023 IEEE/CVF International Conference on Computer Vision (ICCV), 2023, pp. 6927–6938. doi: 10.1109/ICCV51070.2023.00640.

29. İ. Kiliç, E. Orhan, E. Kurtulan, M. E. Özel, and G. Arslan, “Embedded system-based image processing methods for detection of forensic events in CCTV videos,” in International Informatics Congress (IIC 2024), 2024, pp. 29–37.

30. F. Gao, S. You, Y. Ge, and S. Zhang, “Gaussian-based adaptive frame skipping for visual object tracking,” Vis. Comput., 2024, doi: 10.1007/s00371-024-03439-7.

31. Y. Liu and K. D. Kang, “Filtering Empty Video Frames for Efficient Real-Time Object Detection,” Sensors, vol. 24, no. 10, 2024, doi: 10.3390/s24103025.

32. M. Arabboev et al., “Development of a Novel Method for Image Resizing Using Artificial Neural Network,” in In: Zaynidinov, H., Singh, M., Tiwary, U.S., Singh, D. (eds) Intelligent Human Computer Interaction. IHCI 2022. Lecture Notes in Computer Science, vol 13741. doi: https://doi.org/10.1007/978-3-031-27199-1_53.

33. M. Arabboev, S. Begmatov, M. Rikhsivoev, K. Nosirov, and S. Saydiakbarov, “A comprehensive review of image super-resolution metrics: classical and AI-based approaches,” Acta IMEKO, vol. 13, no. 1, pp. 1–8, 2024, doi: 10.21014/acta_imeko.v13i1.1679.

34. M. Arabboev, S. Begmatov, K. Nosirov, J. C. Chedjou, and K. Kyamakya, “Development of a novel method of adaptive image interpolation for image resizing using artificial intelligence,” in IVUS 2022: 27th International Conference on Information Technology, pp. 32–38.