BALKAN JOURNAL OF ELECTRICAL & COMPUTER ENGINEERING, Vol. 13, No. 3, September 2025

Research Article

Improving Face Detection Performance of Compressed MPEG Videos by Using Frame-Independent Scene Change Detection Method

Mehmet Ozdem

***Abstract*—With the spread of computer vision applications, the performance of such applications also became prominent, especially when real-time and near real-time use cases are considered. If not all, many object detection algorithms follow a frame-based search approach, where all frames of the MPEG stream are analyzed sequentially. This drastically increases the computation time and the hardware requirements for such systems. This paper proposes employing a new scene-change detection method to improve object and face detection performance by eliminating the need to analyze every video frame. The method provides a frame-independent approach and does not require decoding and re-encoding of MPEG video. The paper also reports the performance test outcomes to exhibit the proposed approach's value. The findings show that a scene-change detection method enhances efficiency and decreases computational demands. Focusing on frames that show scene changes shows notable advancements in object detection performance.**

***Index Terms*—Face detection, Object detection, Frame-by-frame analysis**

I. Introduction

Using computer vision techniques has become increasingly popular in the last years in various verticals such as entertainment, security, health, and even e-commerce. Human face detection is also considered one of the dominant problems used across many domains. Being a dynamic object and having high variability in its appearance human face detection and recognition is known to be a difficult problem [1]. Thus, these applications lead to high computation time and increased hardware requirements.

In video content, a scene change typically refers to a transition from one distinct visual or thematic segment to another. This can include cuts, fades, dissolves, or any other transitions that signify a shift in content. Identifying these transitions involves analyzing changes in video frames. Detecting scene changes or scene cuts before object detection helps to eliminate the complexity of frame-by-frame analysis thus drastically improving performance. Yet, the overhead of existing scene change detection techniques also contributes to the computational time of the overall process.

The main contributions of this paper lie in the introduction of a novel scene-change detection method that significantly enhances the performance of object detection algorithms, particularly in real-time applications. By minimizing the need for exhaustive frame-by-frame analysis, the proposed approach optimizes computational efficiency and resource allocation, resulting in improved accuracy for both object and face detection. The method is frame-independent, avoiding the complexities associated with decoding and re-encoding MPEG video, thus streamlining the processing pipeline. The results demonstrate that focusing on frames with detected scene changes leads to notable gains in detection performance while reducing computational requirements. Additionally, the research paves the way for future integration of this method into existing computer vision systems and invites further exploration of its effectiveness across various video coding standards and diverse datasets.

This study proposes a two-phased scene-change detection algorithm where the overhead of scene detection in such applications are significantly reduced.

A. Problem Statement

One of the key challenges faced by object detection algorithms is the performance bottleneck caused by the sequential analysis of all frames in an MPEG video stream [2]. This approach leads to increased computation time and higher hardware requirements, which can hinder real-time and near-real-time applications. Yet another challenge is the performance overhead of the current scene-change detection approaches itself [3]. Detecting scene change is solely performed by visually comparing sequential frames. This challenge is more obvious, especially for videos coded with fixed GOP. There is room for improvement in this area.

II. Related Works

Scene change detection is a crucial aspect of video processing, particularly for tasks such as video indexing, summarization, and retrieval. In MPEG (Moving Picture Experts Group) videos, detecting scene changes helps in segmenting the video into distinct scenes, which is useful for a variety of applications including content analysis, editing, and navigation.

There are a significant amount of research studies reporting the performance and accuracy of scene-cut detection methods. Most of these methods can be applied to compressed videos thus need for decoding and reencoding is eliminated. One

Mehmet Ozdem is with the Turk Telekom, Ankara, Türkiye e-mail: mehmet.ozdem@turktelekom.com.tr

Manuscript received Nov 02, 2024; accepted Nov 28, 2024.

DOI: 10.17694/bajece.1577997

Copyright © BAJECE

of the important studies on this area, conducted by Sethi and Patel, suggests statistical hypothesis testing to locate the scene cut points in the video [4]. The paper reports very high accuracy (almost 100%) while performance figures are not reported. The model is based on frame-by-frame comparison which promises high accuracy but the performance is still a question.

Another important contribution in this area is the study conducted by Huang and Liao, where scene changes are classified into abrupt scene changes and gradual scene changes [5]. The study examines the intensity of the image to capture potential fade-out and fade-in scenarios to detect gradual scene changes. The proposed model also distinguishes normal changes that happen in the scene and actual scene breaks. The accuracy of the method is reported to be very high concerning previous studies in this domain however the performance benchmarks are not reported.

Fundamentals of scene-change detection were published by Shahraray in 1995 [6]. Study introduces the concepts such as content-based temporal sampling and frame retention which are still being used and most of the studies conducted in this domain follow this terminology. In addition, there are AI-based solutions that take into account frames in videos and images [7].

III. Background

As part of the study, a two-phased scene-change detection algorithm is employed to identify changes between consecutive frames. This algorithm compares frames based on their size (in bytes) and their visual content to detect scene transitions. Once a potential scene change is detected, the corresponding frame is marked as an index frame for further analysis.

A high-level illustration of the proposed scene-change detection solution is given in Figure 1. The frame selector

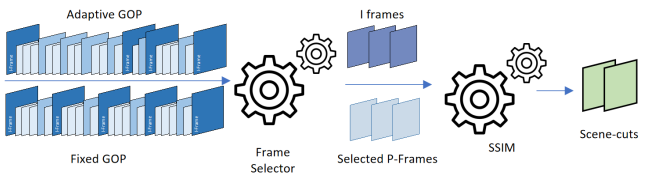

Fig. 1. Building Blocks of Proposed Approach

module identifies if the video is coded with fixed GOP or with adaptive GOP. In case adaptive GOP is used, all I-frames are selected as potential scene-cut frames. In case fixed GOP is used, P-frames close to I-frames in size (bytes) are selected as potential scene-cut frames. All frames marked as potential scene-cuts are then analyzed with SSIM to find the dissimilarity value with the previous I-frame using the SSIM module. MPEG videos are encoded using a series of frames, including I-frames (intra-coded frames), P-frames (predicted frames), and B-frames (bidirectionally predicted frames). Scene change detection generally starts with analyzing these frames:

I-frames serve as key frames containing the complete image data. They are crucial for detecting scene changes because they represent the entire frame independently. P-frames and B-frames depend on preceding or surrounding frames for their data. Scene changes can be harder to detect in these frames due to their reliance on previous frames.

Detection methods often involve comparing I-frames to identify significant differences between them. When the difference between consecutive I-frames exceeds a certain threshold, it suggests a possible scene change.

Advanced Video Coding (AVC) is still by far the most widely used video coding standard for compression and distribution of video content [8]. AVC, also referred to as MPEG-4 Part 10, expresses the video content as a series of frames of various types. Typically, a high-definition (HD) video is composed of 25 frames per second and an ultra-high definition (UHD) video is composed of 60 frames per second.

AVC compression reduces the temporal redundancy across frames using predicted pictures (P-frames) which are coded concerning past full pictures or index frames (I-frame). P-frames contains only the delta concerning the previous I-frame thus, smaller in size. Bi-directional pictures (B-frames) are like P-frames but coded concerning past or future I-frames or P-frames. I-frames or full pictures by nature are heavier in size but still required to be inserted in constant or variable intervals. It is very common to insert an I-frame every 1-2 seconds to enable easier random access, to avoid propagation of errors for longer durations, and to keep the size of frame-to-frame differences at a reasonable level for better compression [9].

This brings us to the concept of a group of pictures (GOP). GOP is the distance between two I-frames expressed in the number of frames. Meaning; for a 25 fps (frames per second) video an I-frame is inserted every 2 seconds GOP is selected to be 50 frames. The notion of GOP is illustrated in Figure 2.

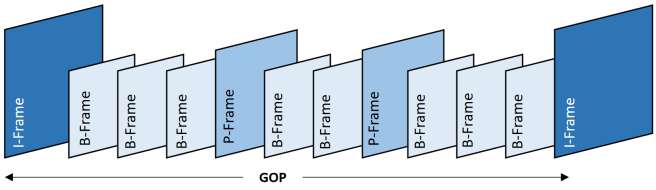

Fig. 2. MPEG Group of Pictures (GOP)

If I-frames are inserted into the video at constant intervals like an I-frame every second or every two seconds, regardless of the scene changes or scene cuts, this is called fixed GOP. A Fixed GOP structure is a static approach where the arrangement of I-frames (intra-coded frames), P-frames (predicted frames), and B-frames (bidirectionally predicted frames) is predetermined and remains constant throughout the video. It offers an easier coding process however introduces the risk of heavier reference frames (B-frames and P-frames) in case there is a scene change within a GOP.

Adaptive GOP, on the other hand, allows for dynamic adjustment of the GOP structure based on the content of the video. The encoder can modify the GOP size and frame types according to scene complexity, motion, and other factors. If not all, most of the decent encoders are capable of coding with adaptive GOP where I-frames are inserted when necessary

[10]. Adaptive GOP promises not only better compression but also improves the chance of identification of scene changes. A comparison of fixed GOP and adaptive GOP approaches is illustrated in Figure 3.

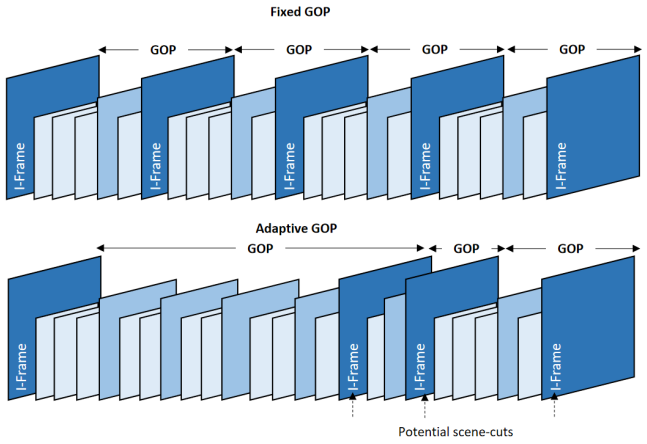

Fig. 3. Fixed GOP vs. Adaptive GOP

Fixed GOP and Adaptive GOP each offer distinct advantages depending on the application. Fixed GOP provides simplicity and predictability, making it suitable for scenarios with consistent content and bitrate requirements. In contrast, Adaptive GOP excels in handling varied content by dynamically adjusting the GOP structure to optimize both video quality and compression efficiency.

As seen, scene change detection is straightforward if the video is coded using adaptive GOP since the assumption of "no scene cut between I-frames" will hold higher chances. Scene detection problem for videos coded with fixed GOP is still a challenge.

In this study, to reduce the complexity of object detection on MPEG videos, a frame selection approach is proposed as opposed to sequential analysis of every single video frame. Frame selection is done based on the detection of scene changes and the sampling of frames for each scene.

The approach is applied to a face detection application. While it is obvious that there will be a performance gain on object detection, the performance drawback of scene-change detection and the increased risk of false-positive results (missing the key object on non-selected frames) are also studied and reported.

IV. Materials and Methods

To address these challenges, we propose a new scene-change detection method that alleviates the need to analyze each frame individually. By identifying scene changes, our method allows for a more efficient allocation of computational resources, thereby improving the performance of object detection algorithms, including face detection. Importantly, our approach is frame-independent, meaning it does not rely on frame-by-frame analysis, and it does not require the decoding and re-encoding of the MPEG video.

The proposed method aims to improve face detection performance by applying a frame-independent scene-change detection approach on compressed MPEG videos. It eliminates the need to analyze every video frame, thereby reducing computation time and hardware requirements. The proposed method aims to improve face detection performance by applying a frame-independent scene-change detection approach on compressed MPEG videos. It eliminates the need to analyze every video frame, thereby reducing computation time and hardware requirements.

As part of the study, a scene-change detection algorithm is employed to identify changes between consecutive frames. This algorithm compares frames based on their size (in bytes) and their visual content to detect scene transitions. Once a scene change is detected, the frame corresponding to the scene change is marked as an index frame for further analysis.

As explained in the previous chapter, detecting scene change for videos coded using fixed GOP is more complex compared to videos coded using adaptive GOP. Therefore, this study is more focused on videos having fixed GOP size.

Noting that P-frames contain only the changes concerning previous index frames a significant increase in size gives us the impression that a scene-change may have occurred. Therefore, the initial phase of the two-phase approach proposed in this study is to detect frame size anomalies in the MPEG frame structure.

Detecting frame size anomalies in MPEG videos involves identifying unusual variations in the size of P-frames (predicted frames) compared to expected values based on typical frame sizes or content characteristics. While there are multiple approaches for anomaly detection a three-step method is preferred in our case:

1) Frame Extraction
2) Statistical Analysis over mean size and standard deviation
3) Anomaly Detection over Z-score method and percentile method

Frame extraction may be done using a variety of tools while open source ffmpeg tool is used in our study. A section from the output of the ffmpeg command where I-frames and P-frames are filtered with their packet sizes is presented in Figure 4. Statistical analysis involves the calculation of average

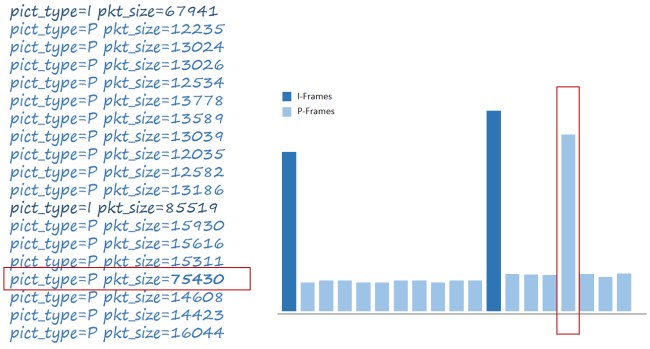

Fig. 4. Video Frames with Type and Size Info

sizes of P-frames and computing the standard deviation to understand the typical variation in frame sizes. Figure 5 presents the implementation of these calculations with Python.

```
frame_sizes = np.array(frame_sizes)

# Compute statistics
mean_size = np.mean(frame_sizes)
std_dev = np.std(frame_sizes)
```

Fig. 5. Implementation of Statistical Analysis

```
# Calculate Z-scores
z_scores = (frame_sizes - mean_size) / std_dev
z_anomalies = np.where(np.abs(z_scores) > z_threshold)[0]

# Calculate percentile thresholds
lower_percentile = np.percentile(frame_sizes, 100 - percentile_threshold)
upper_percentile = np.percentile(frame_sizes, percentile_threshold)

percentile_anomalies = np.where((frame_sizes < lower_percentile) |
                                (frame_sizes > upper_percentile))[0]

# Combine anomalies
anomalies = set(z_anomalies).union(set(percentile_anomalies))
```

Fig. 6. Implementation of anomaly detection

The anomaly detection step employs z-score calculation and percentile threshold together to identify abnormal p-frames. Z-score for each P-frame size is calculated to measure how many standard deviations a frame size is from the mean. As different z-score thresholds are tested, 3 is observed to be the most accurate for the video types used in our case. On top of the z-score, percentile thresholds are also used with a percentile threshold of 95. Figure 6 presents the implementation of these calculations in Python. This approach helps to apply visual comparison only on specific frames rather than traversal of the entire video frame-by-frame.

The scene-change detection algorithm compares identified frames to check significant changes based on visual content. Identified frames are compared with previous frames using structural similarity index (SSIM) calculation [11]. This technique measures the dissimilarity between frames based on pixel intensity differences. A threshold is applied to the dissimilarity measure to ensure that a scene change has occurred. If the dissimilarity value exceeds a predefined threshold, it indicates a significant change between frames, and a scene change is detected.

SSIM is mostly used in the broadcasting industry for measuring the perceived quality of cinematic pictures and digital television content [12]. It simply measures the similarity of the given two images. In this study, SSIM is used to obtain structural dissimilarity of two consecutive video frames. SSIM uses three components to measure the similarity: luminance (l), contrast (c), and structure (s). The formulas to measure these components are as follows.

$$l(x,y) = \frac{2\mu_x\mu_y}{\mu_x^2 + \mu_y^2} \tag{1}$$

$$c(x,y) = \frac{2\sigma_x\sigma_y}{\sigma_x^2 + \sigma_y^2} \tag{2}$$

$$s(x,y) = \frac{\sigma_{xy}}{\sigma_x + \sigma_y} \tag{3}$$

Here, in formula 1, 2, and 3; x represents the pixel sample mean of x and y represents the pixel sample mean of y. x2 represents the variance of x and y2 represents the variance of y. xy represents the covariance of x and y. In formula 4, the overall SSIM value is the weighted product of these components. For simplicity, in this study, we set α, β, and γ values to 1 so all components are equally weighted.

$$SSIM(x,y) = (l(x,y))^{\alpha} \cdot (c(x,y))^{\beta} \cdot (s(x,y))^{\gamma} \tag{4}$$

Finally, the following formula 5 is used to obtain the dissimilarity of given frames, expressed as a percentage. The higher dissimilarity value is the potential scene-change indicator.

$$DSSIM(x,y) = (1 - SSIM(x,y) \times 100) \tag{5}$$

During tests, 90% is used as the threshold since the assumption of dissimilarity above 90% refers to almost completely different images thus a scene cut. Upon detecting a scene change, the frame corresponding to the scene change is selected as the index frame for face detection and subsequent tracking. This frame represents the new scene and serves as a reference for face detection in the scene.

V. Performance Measurement Results

In the scope of this study, performance tests are conducted to evaluate the effectiveness of the proposed approach. These tests are executed on sample videos having various types and durations.

Expected performance directly depends on the type of content. High-motion video content (sports events, action videos) and relatively static content (news programs) have an impact both on scene change detection performance and on face recognition performance. In total 43 test videos containing multiple samples to represent these two content groups were used during tests. Characteristics of sample videos are summarized in Table I. Performance tests are executed

TABLE I
Sample Video Content for Testing

Type	Avg. Duration	Codec and Resolution	Avg. Bitrate	Number of Samples
Action Movie	00:05:10.01	AVC (MPEG-4 Part 10) 30fps, 1080p	2673 Kbps	25
Sports Content	00:04:00.41	AVC (MPEG-4 Part 10) 30fps, 1080p	2430 Kbps	10
News Program	00:11:42.11	AVC (MPEG-4 Part 10) 30fps, 720p	1141 Kbps	8

for two scenarios. Scenario A; where the source video file is provided to the Face Detector module with all frames, scenario B; where the source video file is first provided to the Scene-change Detector module and the Face Detector module is fed only with the selected frames where the scene changes are detected. These scenarios are expressed as block diagrams in Figure 7. Another factor that impacts test performance is the computational capacity of the test environment. Server hosting the Face Detector module is equipped with 2 processors (CPU) both having 2 CPU cores with 3.0 GHz base frequency. The server is also equipped with 64GB of memory.

Scene-change detector module is hosted on a workstation (PC) with a single processor having 10 CPU cores with a maximum of 4.70 GHz base frequency. The workstation is

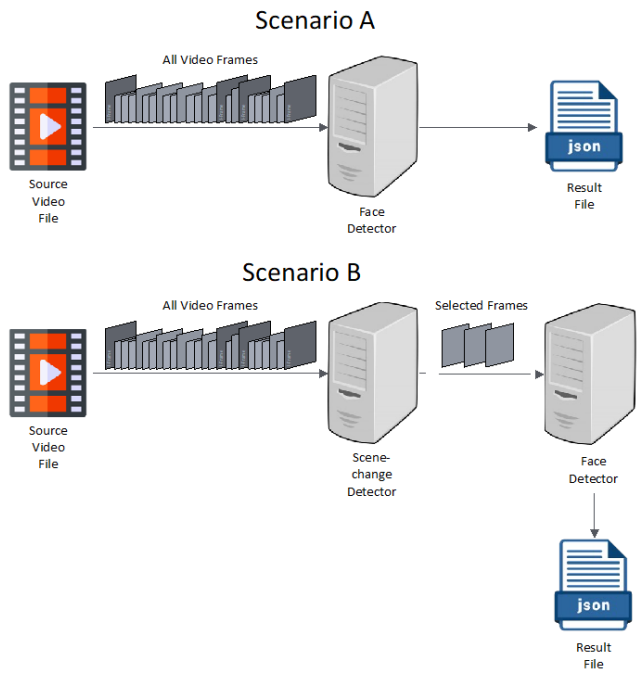

Fig. 7. Test Scenarios

equipped with 32GB of memory. Although it is typical to use Graphical Processing Units (GPU) for such applications, the test in this study is executed on CPU modules for both scenarios. It is assumed that this will have the same impact on both scenarios and thus will not have a significant effect on our benchmarking.

Performance test results are summarized in Table II and Figure 8 for both scenarios. As seen from the test results, despite the overhead of scene-change detection, the scene-change approach significantly improves performance for all content types. The gain is even higher for news programs.

TABLE II
PERFORMANCE TEST RESULTS

Content Type	Average Processing Time for Scenario A	Average Processing Time for Scenario B
Action Movie	997 seconds	393 seconds
Sports Content	894 seconds	307 seconds
News Program	791 seconds	280 seconds

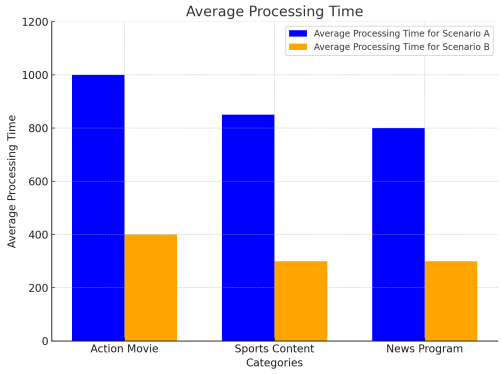

Fig. 8. Average Processing Times per Content Type

The performance gain can also be seen in the number of frames passed to the Face Detector module concerning the total number of video frames. As expressed in Table III and Figure 9, the significant number of frames is eliminated thus the amount of face detection cycles is drastically reduced. These figures also align with the results listed in Table II.

TABLE III
NUMBER OF PROCESSED FRAMES BY FACE DETECTOR

Content Type	Average Processed Number of Frames for Scenario A	Processed Number of Frames for Scenario B
Action Movie	9300 frames	1944 frames
Sports Content	7208 frames	1184 frames
News Program	21084 frames	3051 frames

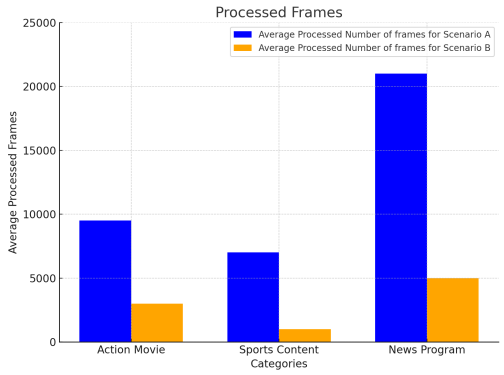

Fig. 9. Processed Frames per Content Type

The objective of the tests was not only to measure the performance gain but also to observe the effect on face detection accuracy. Table IV and Figure 8 summarize the number of face detections per movie type for both scenarios. It is observed that the proposed approach has almost no impact on the number of faces detected.

TABLE IV
NUMBER OF CORRECTLY PREDICTED DATA IN THE TEST RESULT

Content Type	Average Number of Faces Detected for Scenario A	Average Number of Faces Detected for Scenario B
Action Movie	216	211
Sports Content	87	87
News Program	307	300

To summarize; the proposed approach achieves a significantly faster execution time compared to using only the face detector. For instance; for a news program on average the face detection process is completed in 280 seconds, which is around one-third the time taken by the face detector alone (791 seconds). This indicates the efficiency of the proposed method in processing the video frames and performing face detection. Despite the reduced processing time, the approach method maintains a high level of accuracy in face detection.

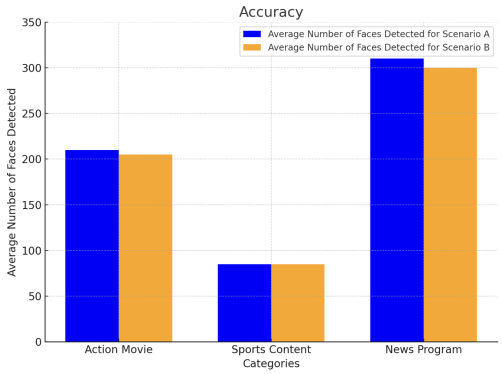

Fig. 10. Accuracy of Proposed Solution

Results highlight the effectiveness of the proposed method in achieving faster face detection while maintaining a high level of accuracy. The optimized processing performance is considered to be beneficial for real-time and near-real-time applications where timely detection is crucial.

VI. Conclusion

The proposed scene-change detection method offers a promising solution for improving the performance of object detection algorithms, particularly in real-time and near-real-time scenarios. By reducing the reliance on frame-by-frame analysis and optimizing computational resource allocation, the approach enhances efficiency while maintaining the accuracy of object detection, including face detection. Future research could explore the integration of the proposed approach into existing computer vision systems and explore its applicability across different domains.

The results demonstrate the improved efficiency and reduced computational requirements achieved by employing a scene-change detection approach. By selectively analyzing frames based on scene changes, significant improvements in object detection performance are observed. It is important to note that the performance of the proposed approach may vary depending on the characteristics of the video, such as scene complexity, motion, and lighting conditions. Further experimentation and evaluation with diverse video datasets are being planned to assess the generalizability and robustness of the method.

Further study is also planned to test the method on different video coding standards such as HEVC, VP9, and AV1. Although AVC is still the dominant standard, these standards are getting more popular and will probably dethrone AVC soon.

References

[1] E. Hjelmås and B. Low, "Face detection: A survey," *Computer Vision and Image Understanding*, vol. 83, no. 3, September 2001.

[2] L. Jiao, R. Zhang, F. Liu, S. Yang, B. Hou, L. Li, and X. Tang, "New generation deep learning for video object detection: A survey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 8, August 2022.

[3] L. Baraldi, C. Grana, and R. Cucchiara, "Measuring scene detection performance," in *Pattern Recognition and Image Analysis: Proceedings of 7th Iberian Conference, IbPRIA 2015*, Santiago de Compostela, Spain, June 2015.

[4] I. Sethi and N. Patel, "A statistical approach to scene change detection," in *Proceedings of SPIE - The International Society for Optical Engineering*, June 1998.

[5] C.-L. Huang and B.-Y. Liao, "A robust scene-change detection method for video segmentation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 12, December 2001.

[6] B. Shahraray, "Scene change detection and content-based sampling of video sequences," in *Proceedings Volume 2419, Digital Video Compression: Algorithms and Technologies*, April 1995.

[7] R. Daş, B. Polat, and G. Tuna, "Recognizing and tracking objects in images and videos with deep learning," *Fırat University, Mühendislik Bilimleri Dergisi*, vol. 31, no. 2, pp. 571–581, 2019.

[8] Bitmovin, "Video developer report 2018," September 2019.

[9] A. Huszak and S. Imre, "Analysing gop structure and packet loss effects on error propagation in mpeg-4 video streams," in *Proceedings of 4th International Symposium on Communications Control and Signal Processing (ISCCSP)*, March 2010.

[10] B. Zatt, M. Porto, J. Scharcanski, and S. Bampi, "Gop structure adaptive to the video content for efficient h.264/avc encoding," in *Proceedings of IEEE International Conference on Image Processing*, September 2010.

[11] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, April 2004.

[12] Z. Wang and Q. Li, "Information content weighting for perceptual image quality assessment," *IEEE Transactions on Image Processing*, May 2011.

Biographies

Mehmet Özdem received his BS degree from Başkent University, Department of Electrical and Electronics Engineering in 2002, his MS degree from Middle East Technical University, Institute of Informatics in 2007, and his PhD degree from Gazi University, Institute of Science, Department of Electrical and Electronics Engineering in 2021. He has more than 20 years of experience in the Telecommunications industry. He worked in many groups (investment, planning, operation, quality assurance, and architecture teams). He currently works as the Network Architecture & Quality Assurance Director and Head of R&D Centers, following responsibilities in Turk Telekom. Simultaneously, he is Vice President of ITU (International Telecommunication Union) SG12 and QSDG organizations. He is also a board member of the Wireless Broadband Alliance. He lectures part-time at universities and has international certificates such as CCIE, PMP, and ITIL.