IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

Cctv Video Storage Optimisation: Motion-Based Frame Removal For Efficiency

Deepak P[1], Vaishnavi BS[2], Yukthi A[3], Hitesh SM[4] , Soumya KN[5]

[1234]*Student*, [5]*Assistant Professor*
[12345] *Department of Artificial intelligence and machine learning*
[12345] *Jyothy Institute of Technology, Bangalore, India*
1

In today's surveillance systems, storing and analyzing CCTV footage is a major difficulty. This research offers a novel and cutting-edge approach to effective storage analysis and optimization by means of motion-based frame reduction methods. Repetitive frames are eliminated by using the OpenCV library, which significantly reduces storage without sacrificing significant motion events. The approach leverages the OpenCV package for Python along with additional technologies like Django, SQLite, and Bootstrap to provide scalability, dependability, and efficient data visualization. Results from experiments have shown how effective various camera configurations in various settings may be at preserving important dataThe method provides a workable way to cut costs, increase analytical effectiveness, and improve usability. This study advances the subject of CCTV surveillance by offering a workable, practically applicable approach. The discoveries open the door for more sophisticated monitoring systems and spur additional advancements in storage optimization.

Keywords: Computer image, picture processing, video analytics, data analytics, motion detection, video compression, storage optimization, and video enhancement.

I. INTRODUCTION

When it comes to security technologies, video surveillance is the most helpful. The storage capacity that the video takes up is one of the trickiest issues with large-scale camera installations [1]. But as the use of these cameras grows exponentially, so does the difficulty of handling and storing the enormous volumes of data (video footage) they produce. Most of the time, this video is kept for extended periods of time or permanently, requiring a large quantity of video storage space consumed, which raises the price. Analysis and the elimination of the pointless films take a lot of time and expensive, professional labor. Numerous strategies have been devised to tackle the obstacles to maximize storage needs and streamline the examination of CCTV recordings. Nevertheless, the current solutions depend on antiquated techniques like manual intervention, scheduled deletion, and video compression [2–3], none of which are effective by today's technological standards. This work introduces motion-footage analysis and storage[4]. To find motion in the video, the method makes use of OpenCV, a powerful and popular open-source computer vision package. The amount of space needed to retain footage is greatly decreased by deleting unnecessary frames, and it is now simpler to get pertinent and crucial information without constantly having to filter out unnecessary data. The method keeps only the most significant sections of the video by deleting frames in which there is little to no motion. An 80% reduction in storage needs can be anticipated for a video with 10% motion 1, which translates into a significant drop in maintenance expenses..

II. LITERATURE SURVEY

A. In today's world, video monitoring is becoming more and more common. Approximately 13 GB of data are generated daily by a single security camera, and since most security cameras are used in multiples, that number might rise to 100 or even 1000 GB each day. Storage space for this material is still a problem due to the massive volumes of data being generated. This section compares and discusses our idea with some of the earlier ones that have been put forth.

As higher compression levels are used, the time-dependent compression technique may, however, have the unintended consequence of lowering video quality for older recordings. This could be harmful as it's possible that you'll need to retrieve old material at any time, and video surveillance becomes less effective if the image quality of the footage is compromised just because it's older.

"A Cloud-Based Storage Optimization Framework" is an additional strategy. The framework consists of a cloud-based video surveillance application, a cloud compute layer, and a cloud storage layer. To maximize storage and management, it makes use of methods including data compression, deduplication, fragmentation, and encryption.

Without the application of machine learning, data compression is unnecessary, while being an excellent tool for tackling security issues.

B. However, as higher compression settings are used, the time-dependent compression technique may inadvertently lead to lower video quality for older recordings. This could be risky since video surveillance becomes ineffective if its quality deteriorates just because it is older, and you never know when you might need to retrieve older footage.

Another tactic is "A Cloud-Based Storage Optimization Framework". A cloud-based video surveillance application, a cloud computation layer, and a cloud storage layer make up the framework. To maximize storage and administration, it makes use of methods including data compression, deduplication, fragmentation, and encryption.

While data compression is a great tool for addressing security challenges, it is not necessary without the application of machine learning.

III. METHODOLOGY

This section outlines the process used to remove frames based on motion from CCTV footage in order to maximize storage. The technique includes the problem definition, software architecture, phases of data processing, algorithms employed, experimental setup, metrics used for assessment, etc.

A. PROBLEM DEFINITION

This study's main goal is to develop software that maximizes CCTV footage storage needs without compromising crucial information. The difficulty is in lowering the amount of storage needed without sacrificing the accuracy and importance of the data that was collected.

B. SOFTWARE MODEL

- The software design of the suggested solution is based on Django, a well-liked lightweight, portable Python web framework. The user specifies how frequently the CCTV camera footage is sent to the server at regular intervals. The system starts the processing pipeline as soon as it receives the film, eliminating any unnecessary frames and storing the video together with the optimized metrics data.

C. PROCESS

For Python image processing applications, the Open Source computer Vision (OpenCV) Library is a well-liked library. It offers a large selection of algorithms for motion detection, facial recognition, object tracking, picture and video analysis, etc. OpenCV is a useful tool in many different applications, including robots, augmented reality, and surveillance systems [7]. It is well-known for its effectiveness and adaptability.

OpenCV offers a widely used method for motion detection called thresholding. Thresholding is a useful technique for dividing a picture or video into the background and foreground according to the pixel intensity.

Applying the same idea, one may determine whether an object is moving by comparing the intensity difference

between successive frames. Pixels exhibiting notable variations in intensity can be designated as foreground, signifying motion, by establishing a threshold value.

The movie.py package is used to change the video extensions in order to guarantee compatibility with different kinds of footage. This ensures that the application can handle data from any type of camera, irrespective of the software being used, the brand of the camera, etc.

When intital video from cameras—regardless of the kind of camera used to record video in any format—is received, it is processed to detect and eliminate still images and motionless frames. As previously indicated, the procedures entail associating the intensity variation across a collection of framing the images. When objects are in motion, they can be identified as being "in motion" by comparing the changes in intensity between successive frames.

It decides whether to preserve or dismiss the frame after recognition. Significant motion would be kept in the video sequence, whereas little or no motion would be recognized as redundant and eliminated. A frame's retention or discarding is determined by the user-specified threshold value. The video's size decreases with a higher threshold value, while its smoothness and integrity increase with a lower threshold value. The balance is determined by the user's particular use case.

D. ALGORITHM USED:

The proposed algorithm adheres to these essential stages.:

- Video Data Retrieval: The total number of frames and frames per second (FPS) are extracted from the supplied video file.
- Video Capture and Initializing: The dimensions of the video frame are ascertained by retrieving the width and height.
- Video Writer Initializing: Frame dimensions, FPS, and the designated codec (compression and decompression methods) are used to generate a video writer object. The optimized video is stored in the output file.
- Redundancy detection and frame processing: Each frame from the input video is read one at a time. For the purpose of making later processes easier, each frame is converted to grayscale. The process of determining the absolute difference between the previous and current frames is known as frame differencing. Significant changes are indicated by a binary mask that is obtained by thresholding the resulting difference frame.
- To determine how much the frame has changed, the amount of white pixels in the mask is calculated. Redundant frames are those that have fewer white pixels than a specified threshold.
- Output Video Generation: The video writer object is used to write non-redundant frames to the output video. By eliminating extraneous frames, the file size is decreased without impacting visual quality.
- Metadata Storage: For future reference, important video metadata is stored in a binary file, including duration and dimensions.

Figure 1: Frames classification

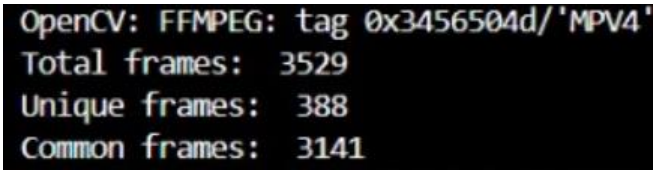
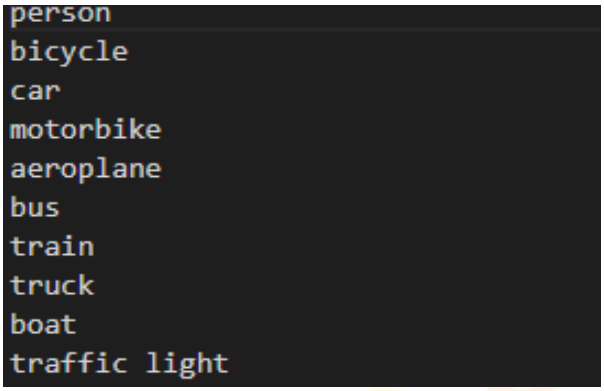


Figure 2 : Recognising objects



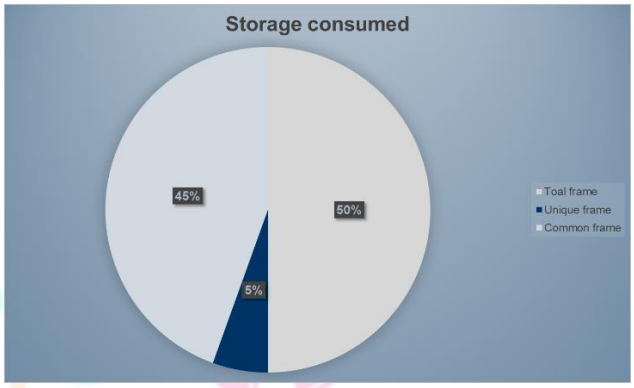
*E. **Pseudocode***

```
#Run
while True:
# Run through the frame
      red, frame = cap.read()
# Check if the loop has reached the end
      if not red:
          break
# Convert the frame to grayscale
grey = ToGrayscale(frame)

# Absolute difference between the present and before frames
difference = AbsoluteDifference(frame,grey)

# Difference to do a binary mask
thresh = threshDiff(differ)

# Number of white pixels
white = WhitePixels(thresh)

# 1000 white pixels and less in the mask, the frame is redundant
if white_pixels < 1000:
continue

# Non-redundant frame to the output video
FrameToOutput(frame)

# Current frame as the before frame for the proceeding iter
prev_frame = grey
# Release Video and writer
releaseVideoCapture_Writer()
```

F. Extras

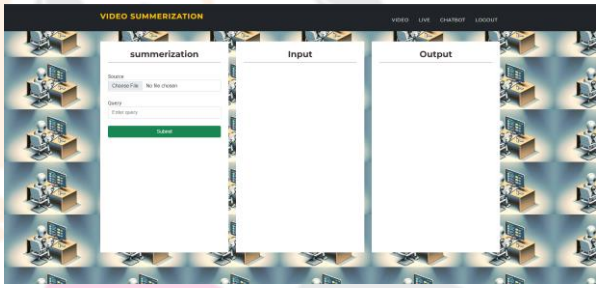
Storage: The final video, free of unnecessary frames, is kept on the server once optimization is finished. An SQLite database holds the metrics pertaining to optimization results, such as the percentage of optimization, motion %, duration, camera information, date, and other pertinent data. Because SQLite is a portable, lightweight, open-source library, it makes it possible to retrieve and analyze stored video and related data quickly.

Figure 3: Storage details



User Interface and UX: A control panel created with Bootstrap presents the optimization results and summary to the users. A visually appealing framework for flexible web design, Bootstrap includes several charts to present the data. Users can make more educated judgments by learning more about the storage optimization metrics thanks to this.

Figure 4 : UI of the tool



IV. TESTING AND EVALUATION

Experiments were done utilizing various camera types in various places with varying angles, backdrops, illumination, and resolution to assess the efficacy of the suggested approach. The security cameras that were utilized had varying characteristics, including resolution, frame rate, and field of vision. They were both indoor and outdoor cameras. The recorded video was saved in its original file formats, such as AVI, MP4, and MOV, which replicated the actual conditions seen in surveillance systems. All of the scenarios saw the algorithm operate successfully, and the variables had little to no impact on the outcome's effectiveness.

On detailed analysis of the particular Algorithm used, it is observed that using this methodology can obtain a 30% reduction in the storage with the standard settings in place.

V. RESULTS

The outcomes indicated that the suggested technique effectively optimized storage.

Videos with a 10% motion rate saw an 80% reduction in storage size without sacrificing any important content. In

order to ensure that important frames are not missed, algorithm 5 demonstrated robustness in identifying a variety of motions, including difficult-to-detect motions like things being tossed, birds flying, and changes in shadow.
However, the user-customizable threshold setting determines the algorithm's efficacy and the smoothness of the resultant films. It is necessary to modify the threshold set to the ideal value in order to strike its best possible balance between preserving major motion events and storage optimization.
A threshold that is set too low may result in limited storage optimization, and a threshold that is set too high may give resultant of the loss of crucial data.

By offering an interface that lets users to adjust the threshold number to meet its individual needs, the technique overcomes this disadvantage. Users may make more informed selections based on their needs thanks to the interface's real-time feedback-loop on the effects of threshold alterations.

VI. Conclusion

This research introduces motion-based frame removal as an effective method for CCTV film analysis and storage. The OpenCV library was utilized to create a user-customizable threshold-based technique, which greatly decreased storage requirements and increased analytical efficiency. In terms of storage optimization, the approach showed encouraging results; for example, a video with 10% motion is predicted to require 80% less storage. In addition, through monitoring significant motion occurrences and removing only the superfluous frames, it guaranteed that no important information was lost all through the tweaking process..
Software architecture, methods, and experimental setting were all described in detail in the methodology section.. Using With Django as a web development system, OpenCV for processing videos, Sql as the database, and Bootstrap for display, an extensible and intuitive system was established.
Trials with several camera kinds in different locations validated its effectiveness and showed how adaptable it is for many types of monitoring scenarios.
While the solution offers significant benefits in terms of optimizing storage and analytic efficiency, it is essential to comprehend its limitations.
The threshold value is essential for maintaining big events of motion while balancing memory decline. To get the desired outcomes, the most suitable threshold value must be chosen based on specific requirements.

In general, study advances in the arena of closed circuit TV video analysis, storage by offering the workable and quality solutions focused on the needs of the industry. The system is appropriate for a variety of surveillance applications because to its user-friendly interfaces, precise motion detection, and effective storage optimization. Both people and businesses who depend on CCTV systems for protection and surveillance can gain a lot from the effort.
Subsequent enhancements to the system's efficiency can focus on automation the parameter selection procedure or implementing customized thresholding techniques that consider surroundings as well as evolving elements, such as the quality of the footage. This will increase effectiveness by applying machine learning's the capacity to strike the perfect equilibrium between maintaining information and storage efficiency.

Additionally, potential improvements are going to concentrate on enhancing the accuracy and productivity of motion recognition, since there is continually potential for improvement through more powerful data and system performance. Finally, this paper illustrated the strategy's applicability in dealing with CCTV footage difficulties involving analysis and storage. We believe that the findings and strategies will lead to further breakthroughs in this field, ultimately leading to more economical and smarter surveillance equipment.

References

[1] Storage Optimization of CCTV Footage Authors: Sonali Shelar | Shweta Gaikwad | Shital Kakade Publication: IJSRD - International Journal for Scientific Research & Development | Vol 8, Issue 1, 2020 | Paper ID: IJSRDV8I10030.

[2] Video Compression Algorithm Based on Frame Difference Approaches Authors: MuzhirShaban Al-Ani | Talal Ali Hammouri Publication: International Journal on Soft Computing | Vol 2, No.4, November 2011.

[3] Time Based Automatic Data Deletion in Role and Policy Based Access Control System Authors: M. Vanitha | C. Kavitha | C. Karthikeyan Publication: World Applied Sciences Journal 34 (3): 307-311, 2016.

[4] Efficient storage and analysis of videos through motion based frame removal. Authors: Akash R, Leandra Shania | 2023

[5] A Cloud-Based Storage Optimization Framework for CCTV Video Surveillance. Authors: A. K. Sahoo, S. K. Patnaik, and B. K. Panigrahi. Publication: Advances in Computing and Data Sciences: 6th International Conference, ICCDS 2021, Coimbatore, India, March 18-20, 2021, Proceedings, Part I, Lecture Notes in Computer Science, vol 12659. Springer, Cham, 2021.

[6] Convolutional Neural Network (CNN) in Keras, towardsdatascience.com/building-a-convolutional-neural-network-cnn-in-keras329fbbadc5f5

[7] X. Cui, Q. Liu, M. Gao, and D. N. Metaxas. Abnormal detection using interaction energy potentials. In CVPR, 2011.

[8] Y. Zhu, I. M. Nayak, and A. K. Roy-Chowdhury. Context-aware activity recognition and anomaly detection in video. In IEEE Journal of Selected Topics in Signal Processing, 2013.

[9] How to Automate Surveillance Easily with Deep Learning, https://medium.com/nanonets/how-to-automate-surveillance-easily-with-deeplearning-4eb4fa0cd68d, 2018.