Scene Change Detection Using Singular Value Decomposition (SVD)

Ramazan Emre Özkaya
150230062
Department of Computer Engineering
Faculty of Computer and Informatics
Istanbul Technical University
Istanbul, Turkey
ozkaya23@itu.edu.tr

***Abstract*—This project aims to detect scene changes with the use of methods like total absolute difference (TAD) and singular value decomposition (SVD). TAD is used as a preliminary step to identify candidate transitions by measuring pixel-wise differences between consecutive frames. SVD is applied to the grayscale matrices of these candidate frames to extract dominant singular values. The Euclidean distance between these singular value vectors is then calculated to validate scene transitions. Results show that the algorithm can identify hard cuts accurately and efficiently.**

Keywords—scene change detection, singular value decomposition, QR iteration, total absolute difference

I. Introduction

Scene change detection is essential in video processing tasks such as segmentation, summarization, and indexing. The most common type of transitions are hard cuts. Hard cuts can be defined as the sudden scene changes where one frame is completely different from the frame after. Accurate detection of such cuts is crucial for efficient video analysis. In this project, we introduce a two-stage approach that integrates TAD and SVD. TAD highlights potential transitions by measuring the absolute pixel differences between successive grayscale frames. To verify the transition, SVD is then applied to these candidates to extract dominant singular values that reflect global frame structure. Euclidean distance between singular value vectors is used to confirm scene boundaries. The method is implemented in Python using OpenCV, with a custom SVD algorithm which is based on QR iteration. Experimental results show that this approach reliably detects hard cuts and can be adapted to various video resolutions.

The method is inspired by the framework proposed by Lu and Shi, which effectively combines pixel-level frame differences with SVD-based feature matching to detect shot boundaries [1].

II. Methods

A. Frame Preprocessing

The video is first read frame by frame using OpenCV. Each frame is converted to grayscale and resized to reduce computational complexity. These grayscale frames are stored as 2D matrices for further processing [2].

B. Candidate Detection via Total Absolute Difference

To detect candidate frames, the Total Absolute Difference (TAD) is computed between each pair of consecutive frames. The computation is as follows [3]:

$$TAD(F_i, F_{i+1}) = \sum_{r,c} |F_i(r,c) - F_{i+1}(r,c)|$$

C. Confirmation via Singular Value Decomposition

For each candidate, the dominant structural information of the frame is extracted using Singular Value Decomposition (SVD) [4].

$$A = U\Sigma V^T$$

To compute the singular values, the algorithm first constructs $A^T A$ and applies QR iteration to approximate its eigenvalues. QR iteration is done as follows [5]:

$$A_k = Q_k R_k$$

$$A_{k+1} = R_k Q_k$$

$$A_k \longrightarrow \Lambda$$

$$V = Q_0 Q_1 Q_2 \cdots Q_k$$

After finding V, each column of U can be calculated as follows:

$$\sigma_i = \sqrt{\lambda_i}$$

$$u_i = \frac{A v_i}{\sigma_i}$$

If σ_i is close to zero, corresponding u_i is set to a zero vector to avoid numerical errors.

D. Comparing with thresholds

Thresholds for both TAD and SVD are determined. In the program, while checking if there is a transition between the frames, the returned values are compared with thresholds. If the returned value is more than the threshold, it is considered as a candidate or a confirmed transition

tad_threshold = 1000000

svd_threshold = 300

(Threshold values were determined after numerous attempts.)

III. Detailed Approach

This part will contain every step of the program with explanations.

At first, the frames from the video are read and the frames are converted into grayscale and resized to 320x180. The frames are put into a list and the TAD values of every adjacent frame is computed and compared with the tad_threshold. If F_i and F_{i+1} has a TAD value more than the tad_threshold, F_i is put into the candidate list. After completing TAD computations, SVD is then applied to the candidates. If the Euclidean distance between the singular value vectors of frames F_i and F_{i+1} exceeds the SVD threshold, frame i is confirmed as a scene boundary. Hence, the program outputs the last frame of each scene, not the first frame of each scene. As a result, there will be no frame from the last scene of the video because there will be no transitions after that scene. Moreover, the confirmed boundary frame is saved as an image and its index is written into a text file. The example will be shown in the Results.

IV. Results

An example was provided by the lecturers. Here are the terminal outputs and other file outputs of the program. The provided example here is video1.mov.

A. Terminal Output

```
Total frames extracted: 846
Candidate transition detected between frame 149 and 150, TAD=3808207 > 1000000
Candidate transition detected between frame 299 and 300, TAD=2609028 > 1000000
Candidate transition detected between frame 449 and 450, TAD=2491535 > 1000000
Candidate transition detected between frame 621 and 622, TAD=2712058 > 1000000
Candidate transition detected between frame 747 and 748, TAD=2712504 > 1000000
SVD distance between frame 149 and 150 = 3997.5445527341317
Frame 149 is a boundary
SVD distance between frame 299 and 300 = 3481.53709445937
Frame 299 is a boundary
SVD distance between frame 449 and 450 = 1278.575352981538
Frame 449 is a boundary
SVD distance between frame 621 and 622 = 481.71139391428534
Frame 621 is a boundary
SVD distance between frame 747 and 748 = 2644.8847261410597
Frame 747 is a boundary
Final confirmed boundaries: [149, 299, 449, 621, 747]
Results saved to 'results/'
```

Fig. 1. Terminal Output

B. File Outputs

1. boundaries.txt

```
1 Detected Shot Transitions:
2 Frame 149
3 Frame 299
4 Frame 449
5 Frame 621
6 Frame 747
```

Fig. 2. The text file that the boundaries are stored

2. Boundary Frames

Fig. 3. Frame 149

Frame 299

Fig. 4. Frame 449

Fig. 5. Frame 621

Fig. 6. Frame 747

As it can be seen, all of the frames are completely different from each other and they are the last frames of each different scene (except the last scene). To further show this, frame 150 and frame 748 will be shown below and the complete transition will be more visible. Frame 150 can be compared with the Frame 149 on the left side.

Fig. 7. Frame 150

Fig. 8. Frame 748 (from the last scene of the video)

These results show that our program is correctly detecting the boundaries between two scenes and saves the images in a folder along with the frame numbers.

V. Discussion

The results show that this method works well for detecting scene changes, especially when the video has clear and sudden cuts. The accuracy depends on the values chosen for the thresholds. If the thresholds are not chosen correctly, the program might miss some transitions or detect extra ones.

This method works best for hard cuts, but it might not detect slow or gradual transitions like fades. Using QR iteration instead of built-in SVD functions makes the program a bit more complex and slower, but it's useful for learning how the algorithm works step by step. In the future, the program could be improved by adjusting the thresholds automatically or by using more advanced techniques.

VI. Conclusion

In this project, an effective method was developed to detect scene changes in videos. The program first uses Total Absolute Difference (TAD) to find possible points where scenes might change. Then, it uses Singular Value Decomposition (SVD) to check if those points are real transitions by comparing the main features of the frames. A custom method using QR iteration was used to perform SVD without relying on built-in libraries. The results show that the method works well in detecting clear scene changes (hard cuts). It correctly finds the last frame of each scene and saves it. The program can also be improved and used in more advanced video applications.

References

[1] Z. Lu and J. Shi, "Fast video shot boundary detection based on SVD and pattern matching," *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, 2013, pp. 1–6.

[2] OpenCV, "OpenCV Documentation," 2023. [Online]. Available: https://docs.opencv.org

[3] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 4th ed. Pearson, 2018.

[4] G. Strang, *Introduction to Linear Algebra*, 5th ed. Wellesley, MA: Wellesley-Cambridge Press, 2016.

[5] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 4th ed. Baltimore, MD: Johns Hopkins University Press, 2013.