Image Processing - Exercise 1

Areen Mansour, areen0507, 212900211

Introduction

a) Goal -The primary aim of this exercise is to develop a scene cut detection algorithm for videos, with a focus on algorithmic concepts rather than specific tools like numpy. The main technique applied involves histogram analysis to identify transitions .between scenes

b) Video Categories

Category1: In these videos, scene changes are expected to be effectively captured by .analyzing the absolute differences in histograms between frames

Category2: Contrarily, videos in Category2 require a different approach. The algorithm adapts to cumulative histogram differences, considering the unique characteristics of .this category

Algorithm

a) Scene Cut Detection Steps-The scene cut detection algorithm involves the following steps:

1. **Video Loading:** Open the video file and initialize necessary variables.
2. **Frame Processing:** Loop through video frames, reading and storing each frame.
3. **Histogram Calculation:** For each frame, calculate the grayscale histogram.
4. **Distance Metric:** Based on the video category, determine the appropriate distance metric.
    - **Category1:** Utilizes the absolute differences in histograms between consecutive frames.
    - **Category2:** Adapts to cumulative histogram differences, considering the entire frame history.
5. **Scene Cut Identification:** Store calculated distances and find the frame index where the scene cut occurs.

b) Modifications for Video Categories-To adapt the algorithm between Category1 and Category2, the key modification lies in the distance metric calculation:

- **Category1:** Measures the instantaneous difference between current and previous histograms.
- **Category2:** Accounts for cumulative differences by comparing cumulative histograms over time.

Implementation Details

a) Algorithm Implementation-The algorithm processes a video frame by frame to detect scene cuts.

- It calculates histograms of grayscale frames for consecutive frames.
- For `video_type = 1`, it measures the absolute difference between histograms of consecutive frames.
- For `video_type = 2`, it calculates cumulative histograms and measures the absolute difference between cumulative histograms.

b) Implementation Source-Implemented from Scratch:

  - Video frame processing, histogram calculation, and scene cut detection logic.
- Iterating through frames, calculating histograms, and handling video file reading.
- Distance metric calculations for both `video_type` cases.
- Used Existing Libraries:
  - OpenCV (`cv2`):
    - Utilized for video file reading (`cv2.VideoCapture`).
    - Grayscale conversion (`cv2.cvtColor`).
    - Histogram calculation (`cv2.calcHist`).
  - NumPy (`np`):
    - Used for numerical operations, array manipulations, and distance metric calculations.

c) Hyper-parameters and Choices:

-None Explicitly Defined:

  - The algorithm does not have explicit hyper-parameters or thresholds set.
  - The distance metric is the primary factor, implicitly adjusting based on video_type.

d) Challenges and Solutions

Challenge:

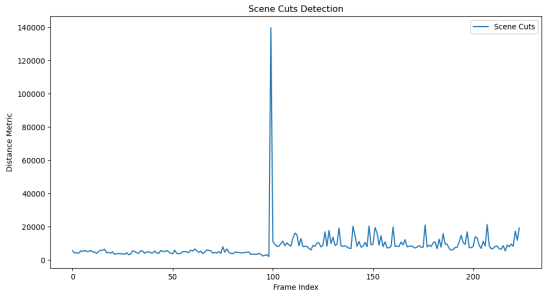
- Ensuring efficient processing of video frames, especially for large videos.
- Handling potential variations in video quality and lighting conditions.

Solution:

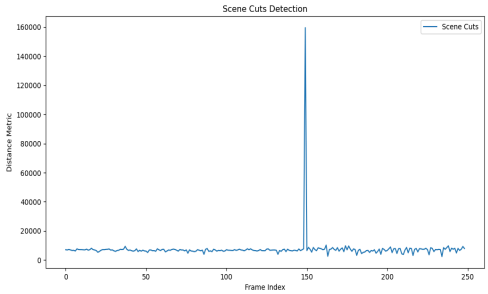
- Implemented a frame-wise processing approach to manage large videos efficiently.
- Relying on grayscale histograms helps in handling variations in lighting conditions

Category 1 Results

For Category 1 videos, the scene cut detection algorithm relies on the absolute difference in histograms between consecutive frames. This means that abrupt changes in pixel intensity and color distribution trigger a scene cut.



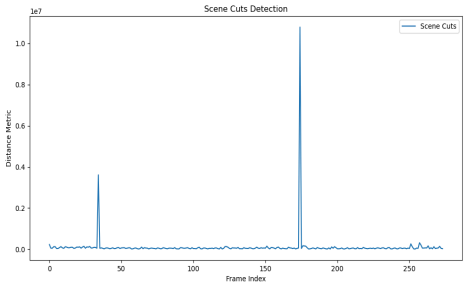
video 1, Scene cuts detected at lst_fr: (99, 100)



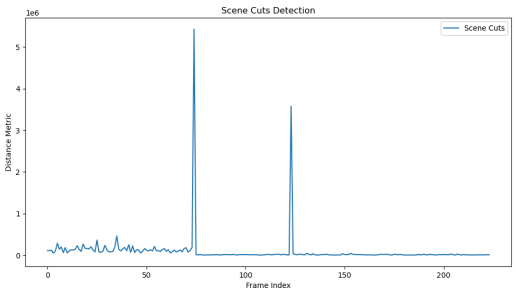
video 2, Scene cuts detected at lst_fr: (149, 150)

Category 2 Results

The scene cut detection algorithm for Category 2 videos employs a cumulative difference approach. This involves computing and comparing the cumulative histograms of consecutive frames, which might be indicative of more gradual changes in the video content.

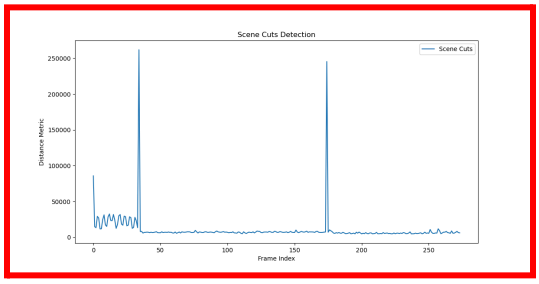


Video 3, Scene cuts detected at lst_fr: (174, 175)

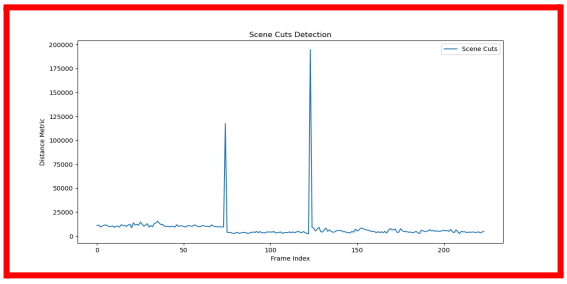


video 4, Scene cuts detected at lst_fr: (74, 75)

(d)



Video 3 category1, Scene cuts detected at lst_fr:(34,35)



video 4 category1, Scene cuts detected at lst_fr: (123, 124)

The algorithm used in the provided code is based on histogram differences for scene cut detection. The explanation of why the algorithm used for video type 1 didn't work well for video type 2 can be understood by looking at the differences in the calculation of the distance metric between the two types.

Conclusion

In conclusion, the algorithm demonstrates adaptability across different video categories through tailored scene cut detection methods. Specifically, employing absolute histogram differences for Category 1 and cumulative histogram differences for Category 2 optimizes performance based on unique video characteristics. The algorithm effectively identifies scene cuts by calculating frame distances, offering valuable insights into transition points. Additionally, the exercise emphasizes the importance of robust implementation in overcoming challenges related to video processing and histogram calculation. Overall, the algorithm's versatility and tailored approaches prove successful in identifying scene cuts, providing valuable insights for diverse video analysis scenarios.