Image Processing - Exercise 1

Yotam Gardosh, yotam.gardosh, 208541334

Introduction

The goal of this exercise was to create an algorithm able to detect the cut between two scenes in each provided video, we were tasked with solving this problem for two categories of videos. The main concept used to solve this problem was the frequency counting histogram of each frame. In order to detect the cut I integrated the histograms all two adjacent frames and compared the sum of the cumulative sums of both, the 2 scenes with the highest difference are the cut.

The difference between the two categories of videos is that category 2 has undergone quantization, since quantization reduces the number of colors or grey levels, two frames from the same scene might change enough so that the corresponding histograms appear from sperate scenes.

Algorithm

1. Convert RGB videos to grayscale.
2. Calculate histograms for consecutive frames.
3. Integrate histogram values.
4. Identify the frame pair with the maximum integrated histogram distance.

Since the algorithm uses the integrated histogram distance as the metrics to detect the cut is solved the problem for both categories, therefore no adjustments were made.

Implementation Details

My implementation of the algorithm described above was done in the following steps:

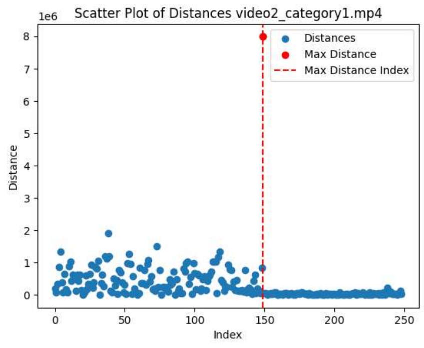
1. To convert RGB videos to grayscale I created a function called “transform_to_greyscale”, this function uses matrix multiplication to shift each frame from RGB to greyscale and return a NumPy array of all the frames in the greyscale video.
2. To calculate histograms for consecutive frames I first create a histogram using a method from the imported NumPy library, then I call a function called “integrate_hist” this function uses methods from NumPy to calculate the cumulative sum of a provided histogram, then it calculates the total integrated value by summing up the cumulative sum and returns that value.
3. To identify the frame pair with the maximum integrated histogram distance I do step 2 for all connecting frames and compare the absolute value of the difference between the integrated sums of both, saving all values, as well as the maximum and the corresponding frames separately.

To analyze the results of the steps above I created a function to explore the data, this function calculates and displays the min, max and mean of the calculated difference between the integrated sums, as well as plots the values with a scatterplot using matplotlib.pyplot.

Given the fact that we were told that all the videos have a scene cut I didn't find it necessary to have a min threshold for the difference between the integrated sums. For cleaner code I saved the greyscale transformer matrix as a global parameter.

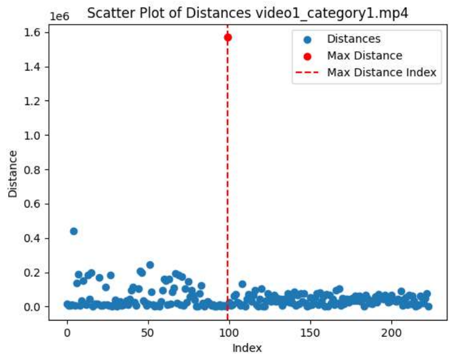
Category 1 Results

Video1:



Statistical Information:
Min Distance: 153.00
Mean Distance: 54906.30
Max Distance: 1568576.00
Index: 99

Video 2:

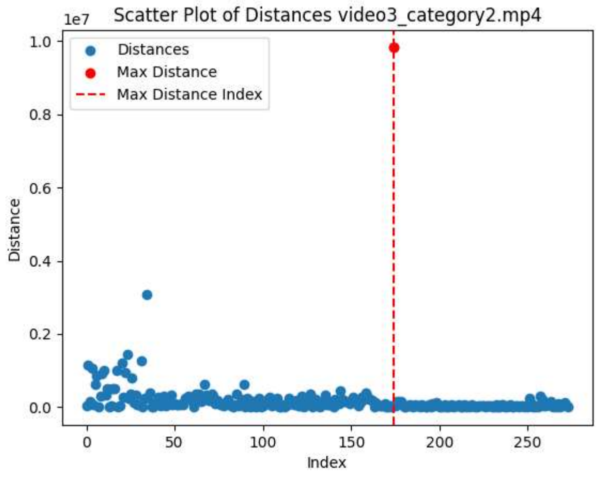


Statistical Information:
Min Distance: 422.00
Mean Distance: 328748.90
Max Distance: 7990172.00
Index: 149

Category 2 Results

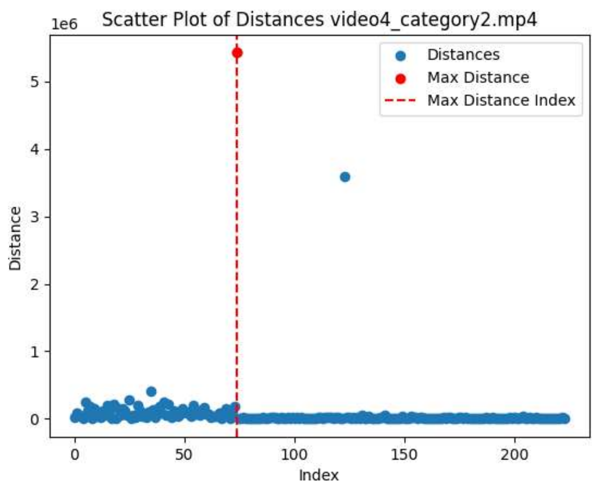
The videos in category 2 have undergone quantization, I was able to detect this first by looking at the provided videos, this was further confirmed by the scatter plot of the difference between integrated sums, we can notice some outliers with higher distance, not as hight as the scene cut, caused by the quantization.

Video 3:



Statistical Information:
Min Distance: 99.00
Mean Distance: 202691.45
Max Distance: 9820298.00
Index: 174

Video 4:



Statistical Information:
Min Distance: 15.00
Mean Distance: 80425.39
Max Distance: 5423722.00
Index: 74

Conclusion

In conclusion the scene cut detection algorithm effectively identifies scene transition in both category 1 and category 2 videos. Is does this by integrating the cumulative sum of each frame’s histogram and comparing the results, as shows in class this provides us with a way to emphasize the difference between histograms. The maximum value represents the exact moment of the cut.