Name: Hari Krishnan Raj Kumar
Student Number: 2374287

CSS 584 Multimedia Database Systems
Autumn 2023
Assignment 4 – Option 2

VIDEO SHOT BOUNDARY DETECTION SYSTEM

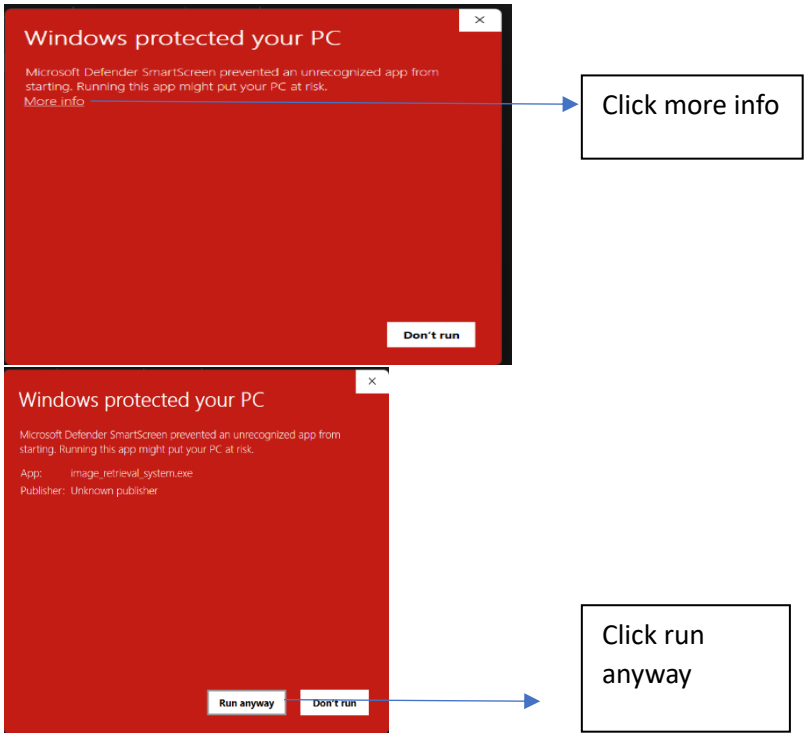
I have chosen the option 2 for my Assignment, this document will walk you through the details on how to run the program (exe recommended), how to use the application, the libraries I have used and the outputs.

How to run the code?

Recommended approach: (run the exe file)

1. Just double click or open the **exe file – ui** (application file)
2. Please Wait for **90 secs to 120 secs** for it to launch (takes less than 30 seconds after the first run), thank you for waiting 😊.
3. Then the application will open

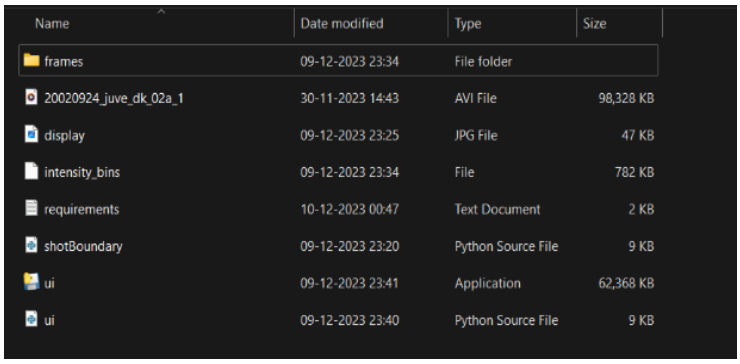
Note: if you windows firewall prevents it from launching please follow as below:



Steps to run the code directly: (to run easily please follow the above approach and not this)

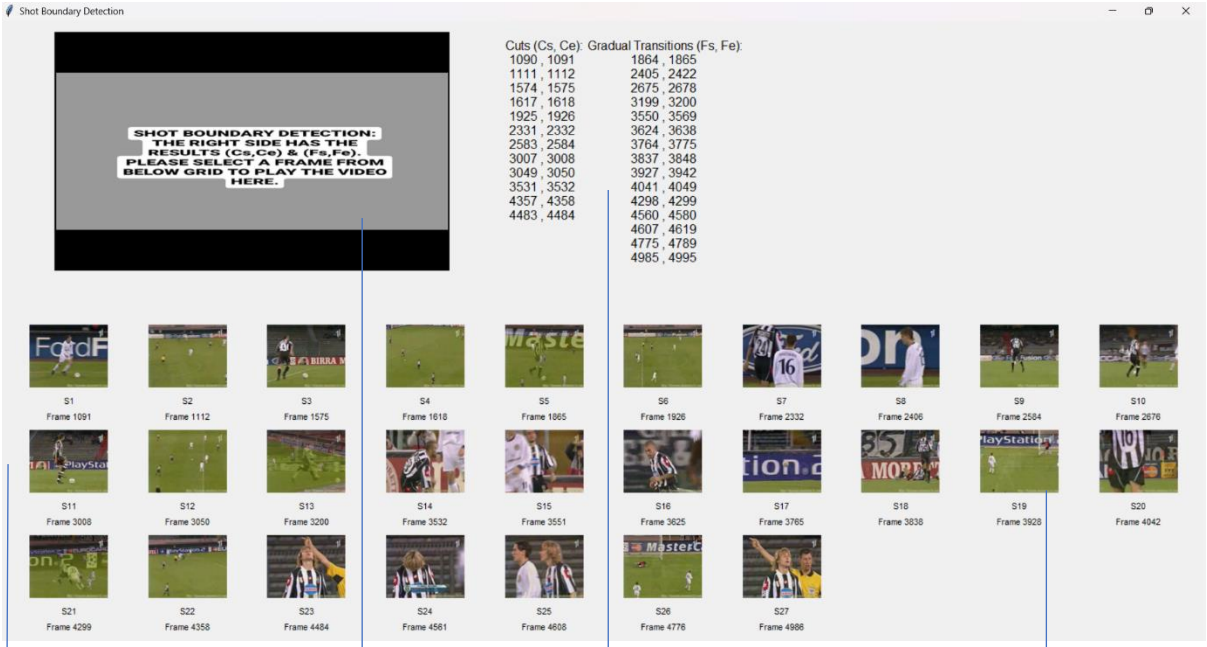
1. Open the Command prompt in the folder
2. Make sure you are in the correct folder.
3. Type “python ui.py” and click enter
4. Wait for >**2.5 minutes** for it to launch

Submission - folder structure:



1. Frames folder: it contains all the frames for the GUI, please not this is used only for the GUI and not for the algorithm. My code creates the frames inside the folder, (if you delete the images inside and run the program you will have a wait time of more than 90seconds for the application to load, please do not delete the folder)
2. 20020924_juve_dk_02a_1.avi: the video file
3. Display: the placeholder in the GUI
4. Intensity_bins: this has computed intensity values for all frames, My code creates this file, it helps in reducing the load time, (if you delete this and run the program you will have a wait time of more than 90seconds for the application to load)
5. Requirements.txt, just in case if you want to run the code, *but the recommended approach is to run the exe.*
6. shotBoundary.py : contains all the logic of my assignment - twin comparison method
7. **ui Application file: Please double click to run this (recommended approach)**
8. ui.py: Contains code of the GUI and calls to the shotBoundary class

How to use the system?



Clickable Image Grid.

This is the **main display frame**, I have added some text for easier understanding, Click any image in the grid below of this main display frame and the video will play based on your click in this area

I added the results in the GUI for easy access to the results. Note these are (Cs,Ce) & (Fs,Fe) and not (Fs+1,Fe)

Click on any image to play based on it.The text below it is the number in the sorted order, abd frame number based on Ce or Fs+1

Once you click on any frame the video will play on the main frame area – Refer next page

Video plays here.

Functionalities of each program file:

ui.py: Contains code related to the GUI, calls functions in shotBoundary class as necessary, this file contains code to create the GUI, populate the grid of frames, play the video on click of images. The GUI uses the images in the frames folder which is created by the code automatically if the images inside the frames folder are deleted. The frames folder itself should not be deleted. But the images inside can be deleted and my code will create them again if its deleted. Note: this is used only in the GUI part and is done to reduce the initial load time

important functionality in this piece of code is how the video is played as the rest of it is just calling functions and creating the GUI

Thumbnail Click in GUI: When a thumbnail representing the frame is clicked, the play_video_from() method is triggered (on click trigger). It sets the selected_index to the index of the clicked frame in the grid.

Video Playback:

1. When a thumbnail representing the frame is clicked, the play_video_from() method is triggered (on click trigger). It sets the selected_index to the index of the clicked frame in the grid.
2. play_video_from() method retrieves the frames range corresponding to the selected frame index.It then iterates through the frames, updating the main display with each frame at a slight delay controlled by root.after().

3. The play_from_frame() method updates the frameLabel with the frames extracted from the video. It uses the after() method to schedule the next frame's display after a short delay (15ms).
4. This delay helps simulate video playback. The process continues until it reaches the end frame of the selected range.

shotBoundary.py: Contains code to calculate intensity bins by opening the video and iterating it for each frame, has a function to extract frames for Gui and store in frames folder. Calculates SD, sets threshold and finds cuts and gradual transition based on the twin comparison method. (The intensity values are read from the intensity bins file, if you delete it my code will generate it again, but this will increase the initial load time.)

Methods for Intensity Values
check_intensity_values(): Checks if the intensity values file exists. If not, it generates intensity values.
load_intensity_values(self): Loads intensity values from a file if it exists.
create_intensity_values(self): Creates intensity values by calculating histograms of frames within a specified range and saves them to a file.
generate_sd(): Generates SD (Sum of Absolute Differences) values between consecutive intensity histograms.
set_thresholds(): Sets threshold values (ts and tb) based on SD values.

How cuts and gradual transitions are found?

Find_frames():

Iterates through the pre-calculated SD values between consecutive intensity histograms of frames.
Identifying Cuts:

1. When the SD value exceeds a higher threshold (self.tb), it indicates an abrupt change, marking a potential cut.

2. Records the cut start (cs) and cut end (ce) indices and stores them in the frame_results dictionary.

Detecting Gradual Transitions:
For SD values in the range range (self.ts to self.tb),

1.Start exploring frames to identify the end of the gradual transition.

2.When conditions (tor == 2)is met or cut is detected, it marks the end of the gradual transition.

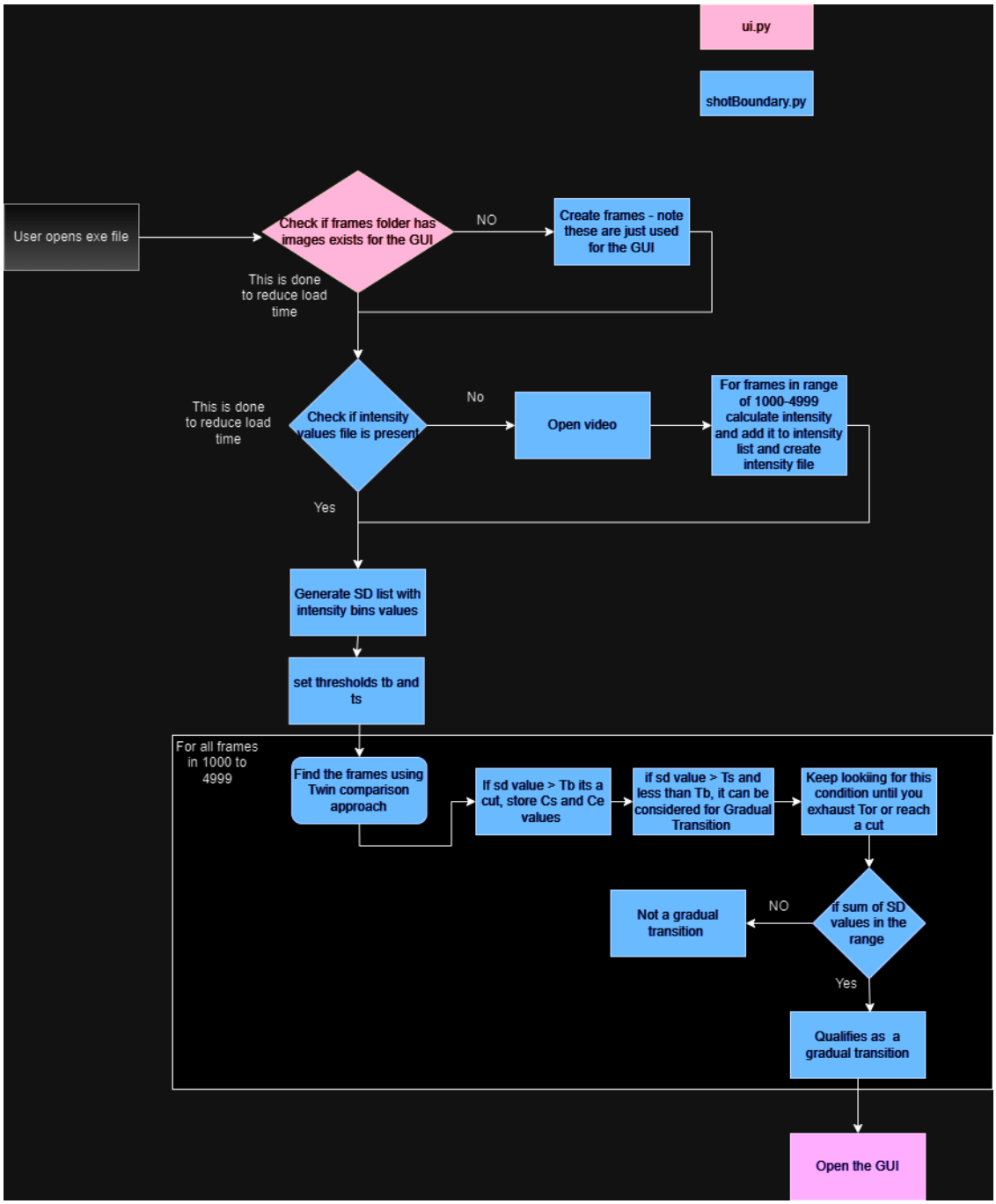
Records the gradual transition start (fs) and end (fe) indices in the frame_results dictionary by checking the through the sum of sd values >Tb using the util function.

3.Updates the next_frame_index to skip frames already processed for cut or gradual transition detection

Find_frames_uti():

1. Checks the sum of SD values within a specified range (fs_candi to fe_candi) to validate if it is a gradual transition.
2. If the sum exceeds a threshold (self.tb), confirming the gradual change, it records the start and end indices (fs, fe) of this transition in the frame_results dictionary.

I have explained this better with the flow diagram in the next page:



Libraries, Tools and Techniques:

- I have used python for my assignment.
- Tkinter for the GUI
- I have used opencv library for opening the video, intensity calculations
- Numpy library for all the calculations especially for intensity and SD calculations.
- Pillow for loading images into the GUI.

- I have saved the frames in frames folder just for the GUI, if the images in the frames folder is deleted the code will generate them again but it will take time, but the *frames folder itself should not be deleted.*
- I have saved the intensity values in a binary file, if it's deleted the code will generate again but will increase the load time
- I am using classes in python, the ui class has functions related to the GUI where as the shotBoundary class has mostly the algorithm functions and one function which extracts frames for the GUI.
- I used numpy in most calculations.The rest of the logic is as shown in the flow diagram In the previous page

Output:

'cs': [1090, 1111, 1574, 1617, 1925, 2331, 2583, 3007, 3049, 3531, 4357, 4483],

'**ce**': [1091, 1112, 1575, 1618, 1926, 2332, 2584, 3008, 3050, 3532, 4358, 4484],

'fs': [1864, 2405, 2675, 3199, 3550, 3624, 3764, 3837, 3927, 4041, 4298, 4560, 4607, 4775, 4985],

'fe': [1865, 2422, 2678, 3200, 3569, 3638, 3775, 3848, 3942, 4049, 4299, 4580, 4619, 4789, 4995]}

Start frames:

[1091, 1112, 1575, 1618, 1865, 1926, 2332, 2406, 2584, 2676, 3008, 3050, 3200, 3532, 3551, 3625, 3765, 3838, 3928, 4042, 4299, 4358, 4484, 4561, 4608, 4776, 4986]

End frames:

[1111, 1574, 1617, 1864, 1925, 2331, 2405, 2583, 2675, 3007, 3049, 3199, 3531, 3550, 3624, 3764, 3837, 3927, 4041, 4298, 4357, 4483, 4560, 4607, 4775, 4985, 4999]

Pairs / frame ranges:

[(1091, 1111), (1112, 1574), (1575, 1617), (1618, 1864), (1865, 1925), (1926, 2331), (2332, 2405), (2406, 2583), (2584, 2675), (2676, 3007), (3008, 3049), (3050, 3199), (3200, 3531), (3532, 3550), (3551, 3624), (3625, 3764), (3765, 3837), (3838, 3927), (3928, 4041), (4042, 4298), (4299, 4357), (4358, 4483), (4484, 4560), (4561, 4607), (4608, 4775), (4776, 4985), (4986, 4999)]

Frame issues in accuracy (=5):

3276 & 4892 are missing in Gradual transitions.

Frames different by 1: 3625 instead of 3624, 4299 instead of 4300

4608 instead of 4604