Video Editing Curve Tool

Prepared by
Michael Pérez

Computer Vision
CAP 4410

18 September 2019

Florida Polytechnic University

Table of Contents

List of Figures

1. Overview

1.1 Introduction

My name is Michael Pérez, and I am a senior studying Computer Science at Florida Polytechnic University. This objective of this project was to develop a curve tool for video editing that works just like Adobe Photoshop's curve tool:

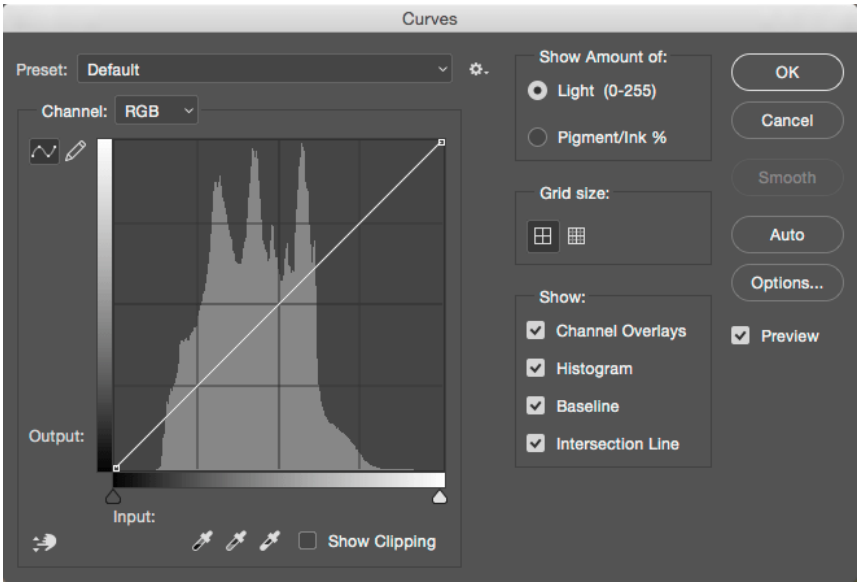

Figure 1: Adobe Photoshop Curve Tool

This tool is used in Adobe Photoshop to adjust the brightness and contrast of an image. The user can add points to the graph, and the line will curve to fit all the points. The resultant curve acts on the histogram of the image. If a point on the line is higher than it was before it was edited, the corresponding shade will be brightened. If a point on the line is lower than it was before it was edited, the corresponding shade will be darkened.

I used OpenCV to achieve operations like file I/O and histogram mapping. I also designed a preview interface where the user can see the effect of a certain curve on the first frame of the video, continuously, as points on the curve are added, removed, and moved. I added a histogram for this window that adjusts continuously as well. The edited video saves in one test case but does not save in any other test case. I will explain this in the Testing section.

1.2 Running the Project

There are six windows that appear when the program is run:

1. Video Editing Curve Tool
2. Video
3. Video Histogram
4. Preview Interface

5. Preview Histogram

While the program is running, pressing the following buttons correspond to these commands:

‘Space Bar’: Play the video

‘P’: Apply the Curve Tool to the video and replace the original video file

1.3 Displaying the Histogram

The histogram of an image displays the frequency of pixel intensity values. The x-axis is the gray-level intensities and the y-axis is the frequency of those intensities:

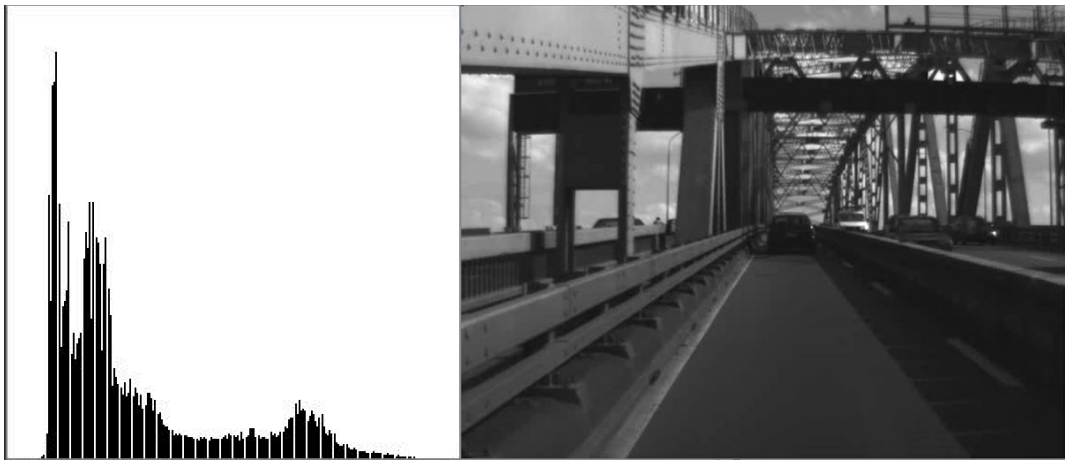

Figure 2: Frame 1 of barriers.avi and Histogram

1.4 Contrast Increase and Decrease

The definition of contrast is the difference between the minimum and maximum pixel intensities in an image. To increase contrast, the darker tones can be darkened, and the brighter tones brightened. Implementing this in the curve tool creates an S and increases the range of the histogram of the image:

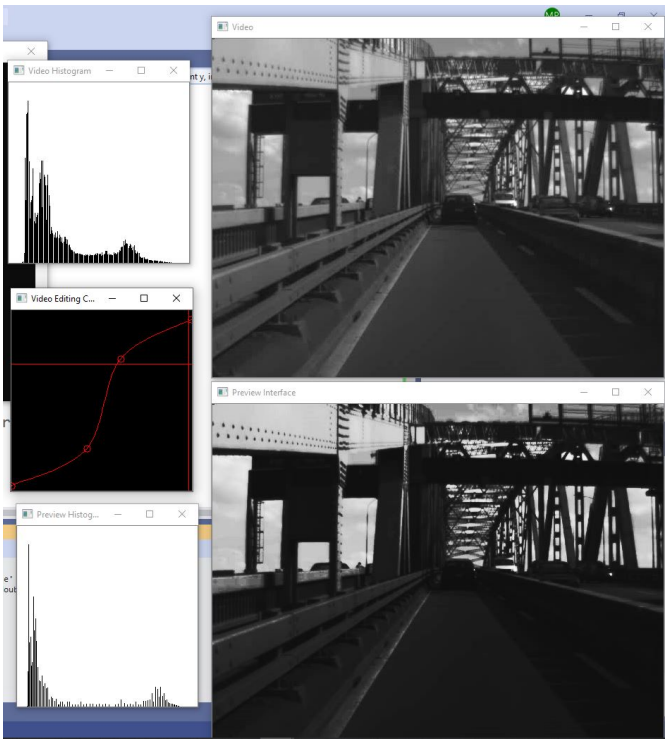

Figure 3: Contrast Increase

To decrease contrast, the darker tones can be brightened, and the brighter tones darkened. The range of the histogram of the image decreases:

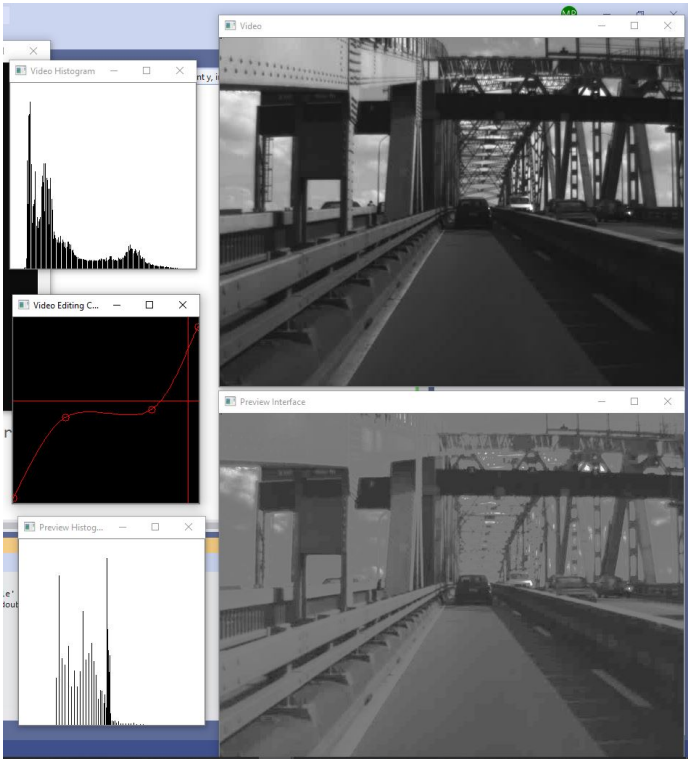

Figure 4: Contrast Decrease

1.5 Brightness Increase and Decrease

Brightness is a relative term and can be defined as the amount of energy output by a source of light relative to the source we are comparing it to. Brightness can be increased by adding the same number to all image pixel values, which brightens all tones. This curve tool does not add the same number, but instead adds a range of numbers. The histogram of the frame shifts right:

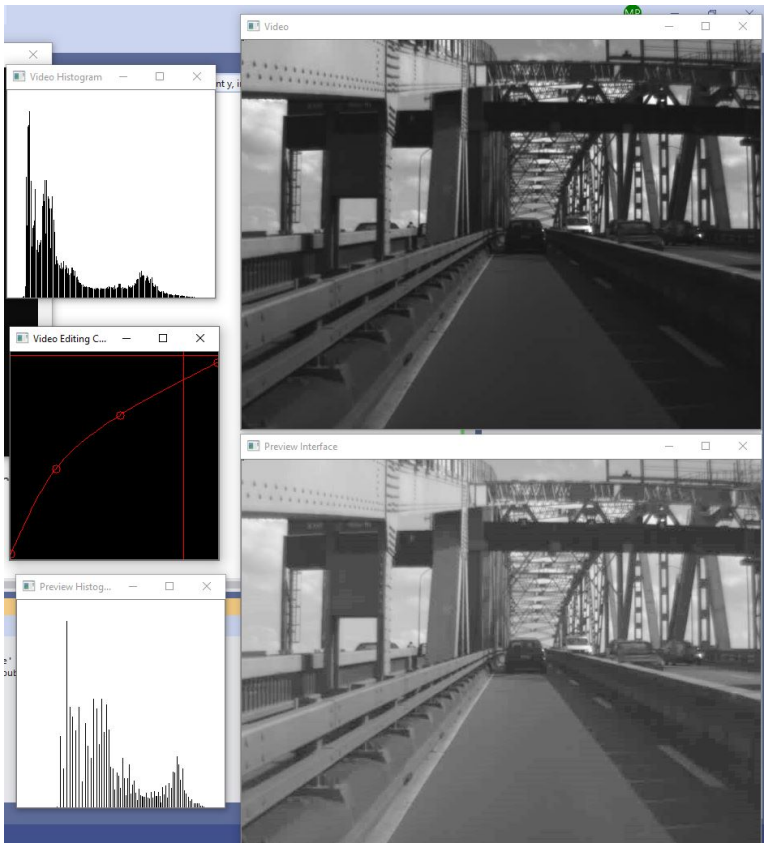

Figure 5: Brightness Increase

Brightness can be decreased by subtracting the same number from all image pixel values, which darkens all tones. This curve tool does not subtract the same number, but instead subtracts a range of numbers. The histogram of the frame shifts left:

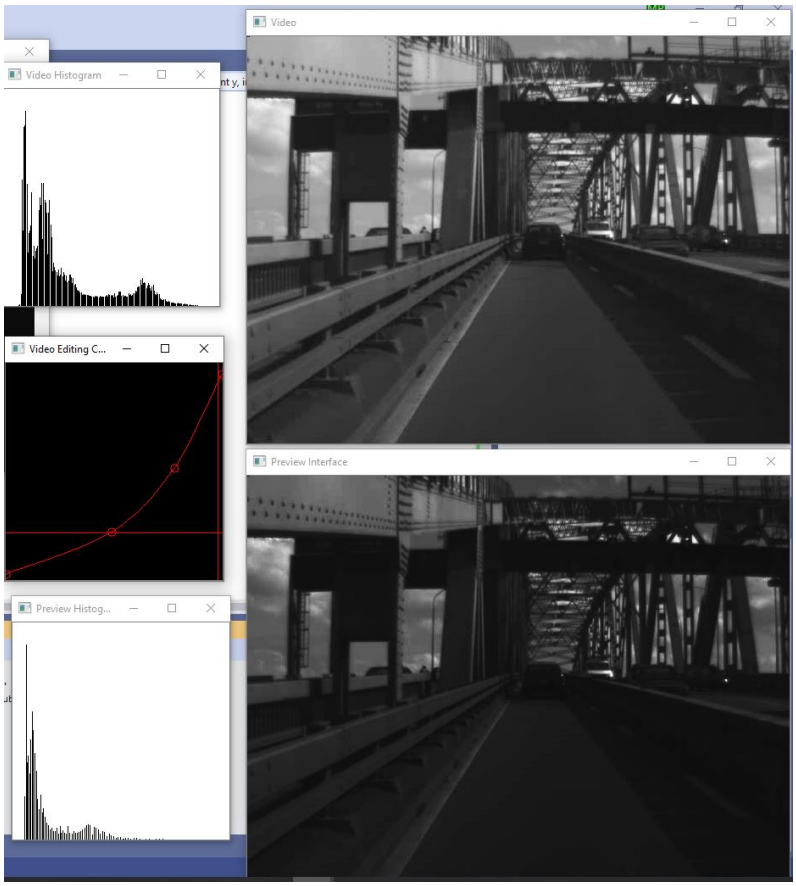

Figure 6: Brightness Decrease

1.6 Product Scope

The Video Editing Curve Tool has the capability of previewing the effect that a curve has on a video. Then, it can save the edited video as the original video file and replace it, in one test case. Once the bug in the code is fixed so that it can successfully save the video in more test cases, this project could be useful for photographers and videographers. It gives them the ability to adjust the brightness and contrast of their videos just like they do with their images in Adobe Photoshop.

2. Implementation

2.1 Playing the video

I read Chapter 11 of source [1], *OpenCV Computer Vision Application Programming Cookbook, 2nd Edition*. I modified the program in the textbook to play the video that the user wants to edit in the Video window. I implemented a loop to continuously check what key the user enters so that the user can press the space bar to play the video and the 'P' button to apply the Curve Tool edit to the video.

2.2 Displaying the Histogram

I read Chapter 4 of source [1] and included the Histogram1D.h class header file in my main.cpp file [4]. Then, I called a function from in that file called getHistogram(). This calls an OpenCV function calcHist(), which returns the histogram of an image. The histogram of the video is continuously updated as the video plays. I did this by generating the histogram within the loop that displays frames of the video.

2.3 Implementing the Curve Tool

I researched methods of implementing Adobe Photoshop's Curve Tool all over the internet and found [3]. This is a page on Stack Overflow that describes the implementation of three Curve Tools that act through three channels on color images:

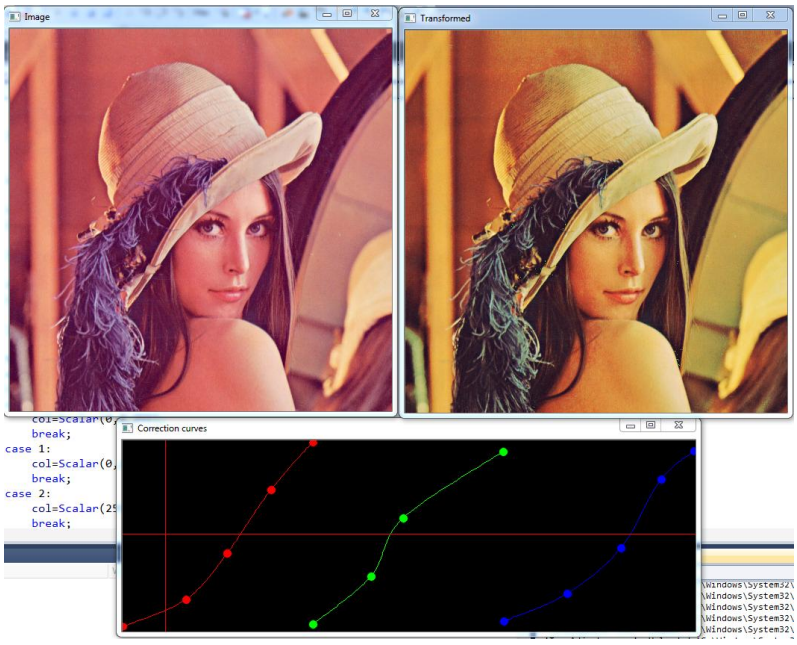

Figure 7: Color Curve Tool

I went to the referenced website, [2], and found the required header files: overhauser.hpp, overhauser.cpp, and vec3.hpp, and included these files in my solutions folder. I modified the code substantially, removing two channels so that only one channel act on the one channel of gray images. I also added functionality; the preview of the edited first frame of the video updates continuously as points are added, removed, and moved in the curve tool window. I did not modify the VideoProcessor.h, overhauser.cpp, overhauser.hpp, and vec3.hpp files.

2.4 Saving the Edited Video

For this stage, I read Chapter 11 of Source [1] and used the Video Processor class (in VideoProcessor.h) [5]. Although saving the file never worked well, this file simplified the process of reading, processing, and writing a video by providing functions for me to use in the main program. I also used some lines of code from the videoprocessing.cpp file. I performed the video processing similarly, but my processing method applies the lookup table to the image based on the Curve Tool.

3. Testing

3.1 Successful Test Cases

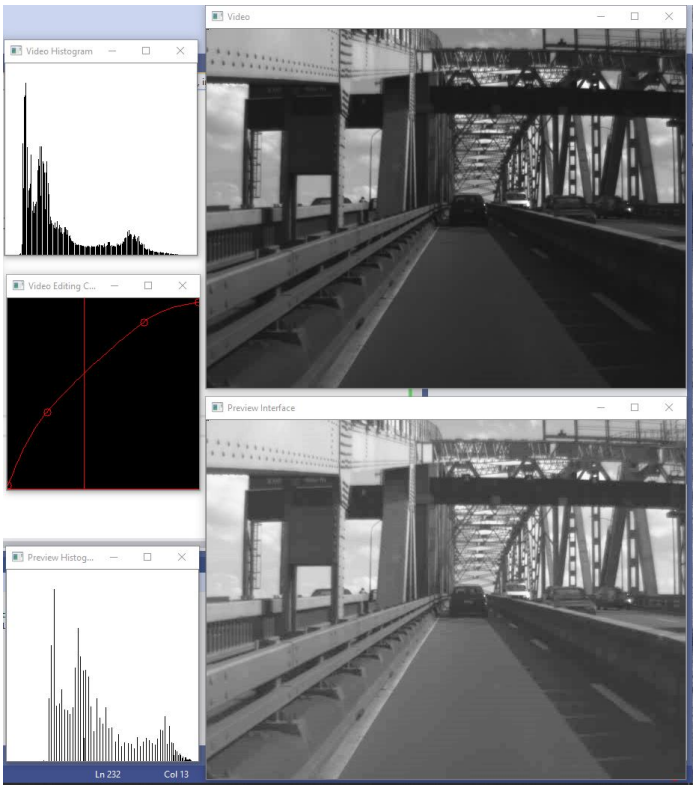

Figure 8: Test Case 1

This curve increases the brightness of the image above, as expected.

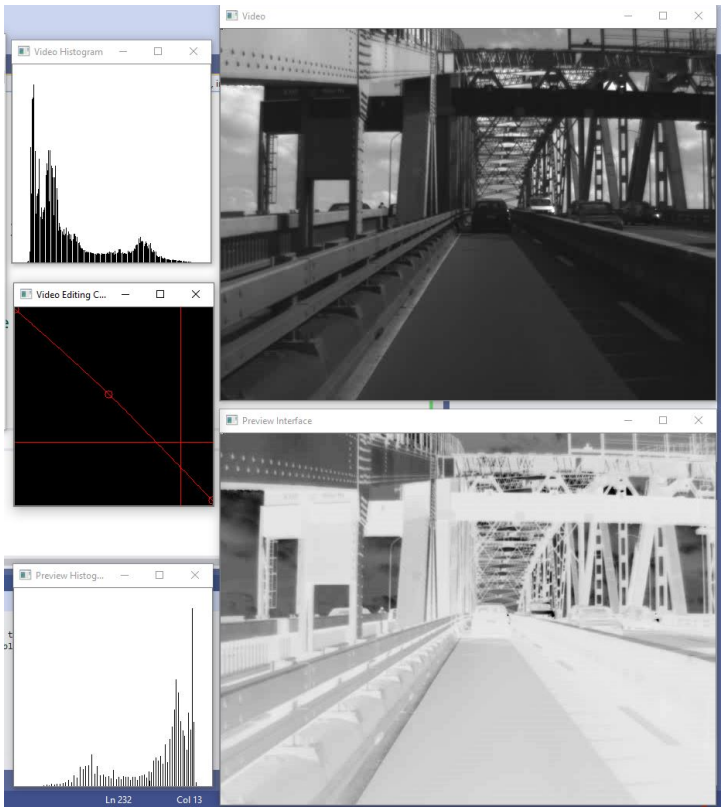

Figure 9: Test Case 2

This curve inverts the color in the image above, as expected.

3.2 Saving the video

All aspects of the project function as desired except for the ability to save the edited video. In this test, the video processing function that the VideoProcessor object calls applies the Curve Tool to the image preview:

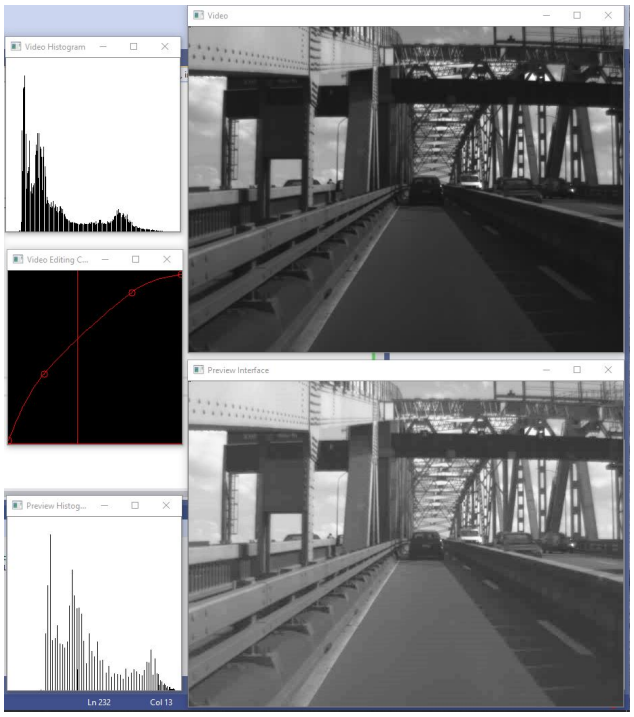

Figure 10: Test Case 3a

When ‘P’ is pressed to apply the Curve Tool effect to the entire video, this is outputted:

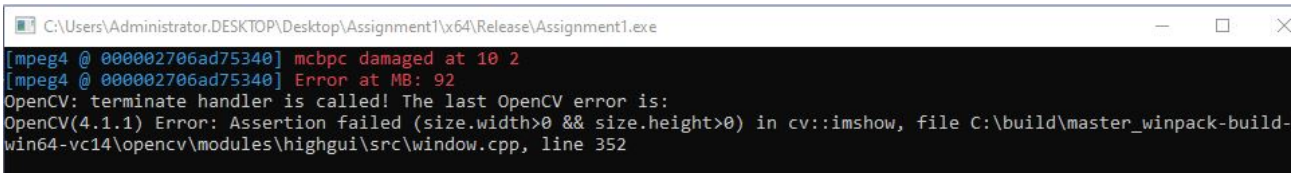

Figure 11: Test Case 3b

It appears that the Writer object tries to write the edited file, then the file becomes corrupted for some reason. I cannot find any solution to this error online.

In the next test case, I change the video processing function to calculate the Canny edges of the image like the original processing function in [5] does. Once two or more points are placed on the curve tool, the function is applied to the image:

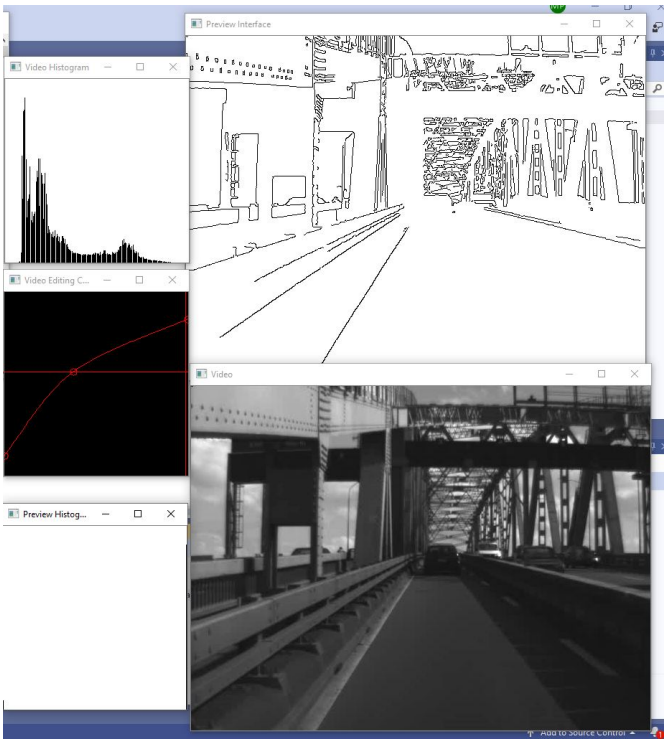

Figure 12: Test Case 4a

When ‘P’ is pressed to calculate Canny edges of the entire video, this occurs:

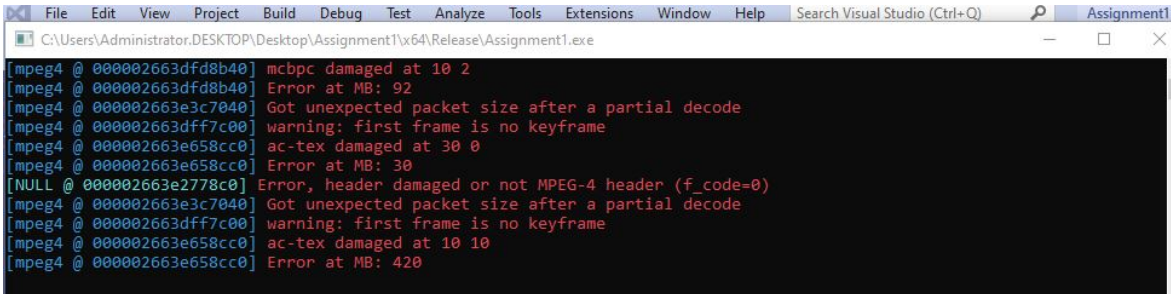

Figure 13: Test Case 4b

Then, it continues reproducing the same errors:

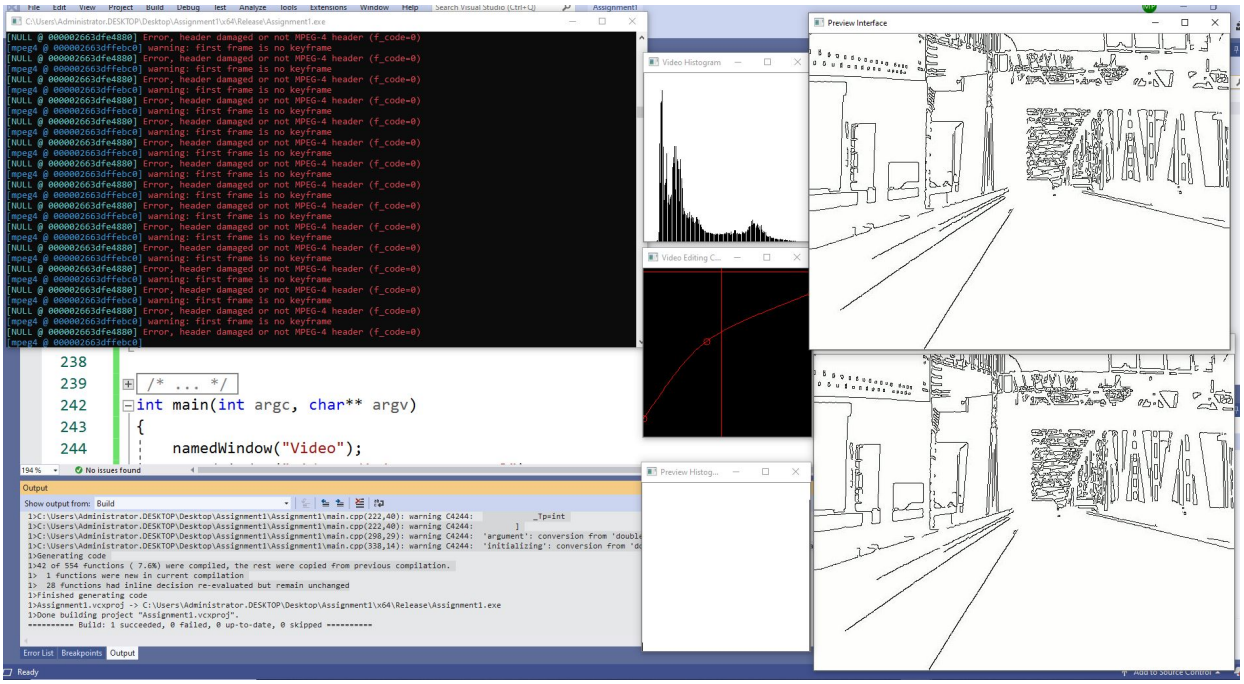

Figure 14: Test Case 4c

Then, the program replaces the original file with this video of the canny edges of the frame. The program successfully replaces the original video with the edited video. The first frame of the Video window changes to the edited video of canny edges, like it should. Then the program shuts down. When the program is run again, the canny edges video can be played for two seconds:

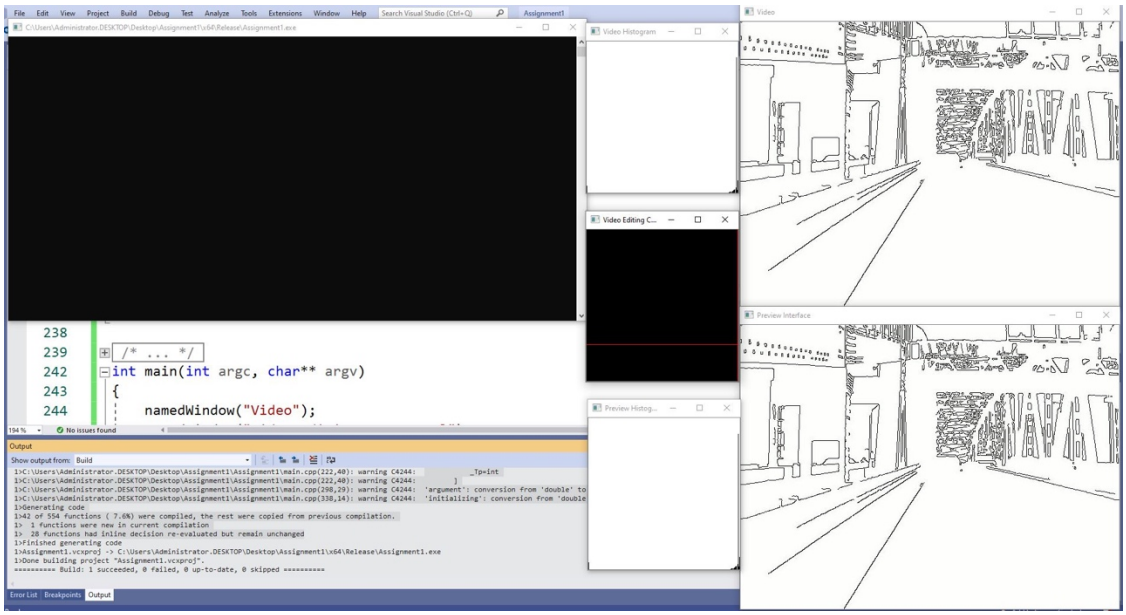

Figure 15: Test Case 4d

References

[1] R. Laganiere, *OpenCV Computer Vision Application Programming Cookbook*. Birmingham, U.K.: Packt Pub., 2014.

[2] R. Gruian, "Overhauser (Catmull-Rom) Splines for Camera Animation", *Codeproject.com*, 2019. [Online]. Available: https://www.codeproject.com/script/Articles/ViewDownloads.aspx?aid=30838. [Accessed: 19- Sep- 2019].

[3] A. Smorodov, "how to draw curve on control points using opencv", *Stack Overflow*, 2019. [Online]. Available: https://stackoverflow.com/questions/23641208/how-to-draw-curve-on-control-points-using-opencv. [Accessed: 19- Sep- 2019].

[4] "PacktPublishing/OpenCV3-Computer-Vision-Application-Programming-Cookbook-Third-Edition", *GitHub*, 2019. [Online]. Available: https://github.com/PacktPublishing/OpenCV3-Computer-Vision-Application-Programming-Cookbook-Third-Edition/blob/master/Chapter12/histogram.h. [Accessed: 19- Sep- 2019].

[5] "PacktPublishing/OpenCV3-Computer-Vision-Application-Programming-Cookbook-Third-Edition", *GitHub*, 2019. [Online]. Available: https://github.com/PacktPublishing/OpenCV3-Computer-Vision-Application-Programming-Cookbook-Third-Edition/blob/master/Chapter12/videoprocessor.h. [Accessed: 19- Sep- 2019].