CMPT 365 Final Project Report

By Maoshun Shang (301280479)

Chang Liu (301294110)

Instructor: Mark Drew

Computing Science Department, Faculty of Applied Science

Simon Fraser University

Introduction:

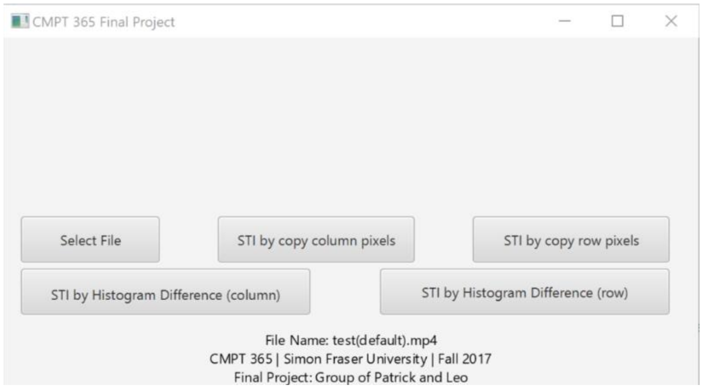
Spatio-temporal image (STI) is the image that consists of video segments (columns or rows) for each frame through time scale.

Language used: Java

Library used: OpenCV, JavaFX, Java Utility.

Functions:

For this final project, we have designed the GUI through JavaFX application. As shown in Fig. 1, there are five buttons below.


CMPT 365 Final Project
Select File
STI by copy column pixels
STI by copy row pixels
STI by Histogram Difference (column)
STI by Histogram Difference (row)

File Name: test(default).mp4
CMPT 365 | Simon Fraser University | Fall 2017
Final Project: Group of Patrick and Leo

Fig. 1: the user interface base

Users could click "Select File" button to choose an anticipated file. If no file selected, the program has default video so that it prevents runtime errors in the following steps.

The "STI by copy column pixels" button is designed for copying the center column pixels of a video. We can construct STI through the composition of center column pixels in every frame.

Ditto on "STI by copy row pixels" button, the STI which based on every center row of the frames will be shown on the image view.

The "STI by Histogram Difference (column)" button is designated to construct the STI that made from histogram intersections of each column in each frame with previous time instant. Take the default video as an example, there is a straight line with a slope regarding the speed of the transaction. By the method of histogram intersection, the STI made by histogram intersection could be easily found.

The "STI by Histogram Difference (row)" button takes the same scheme as the column but based on histogram difference by rows. "STI by Histogram Difference (row)" button constructs the STI through the intersection of each row.

General:

1.1. STI by Copying Pixels

After running the program, the application windows (Fig.1) will show on screen.

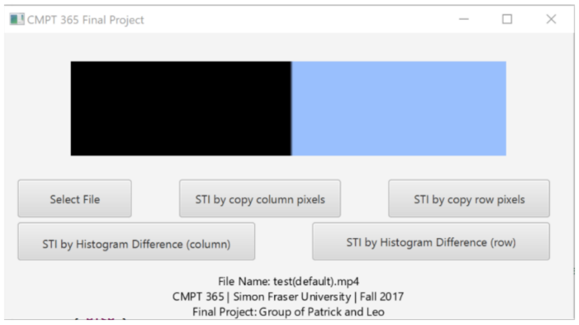

CMPT 365 Final Project
Select File
STI by copy column pixels
STI by copy row pixels
STI by Histogram Difference (column)
STI by Histogram Difference (row)
File Name: test(default).mp4
CMPT 365 | Simon Fraser University | Fall 2017
Final Project: Group of Patrick and Leo

Fig.2(a)

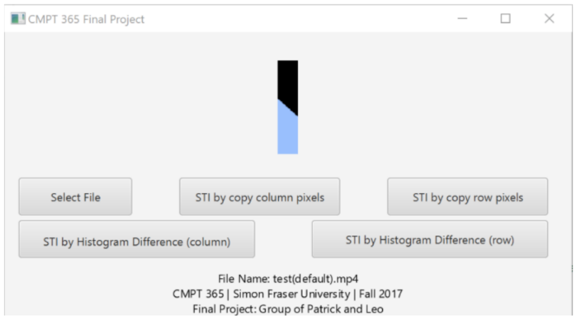

CMPT 365 Final Project
Select File
STI by copy column pixels
STI by copy row pixels
STI by Histogram Difference (column)
STI by Histogram Difference (row)
File Name: test(default).mp4
CMPT 365 | Simon Fraser University | Fall 2017
Final Project: Group of Patrick and Leo

Fig. 2(b)

When started by default test video, click "STI by copy column pixels" button. You will obtain the image like Fig. 2(a) within few seconds. When you click "STI by copy row pixels" button, the screen will show the image, like Fig. 2(b).

1.2. STI by Histogram Differences

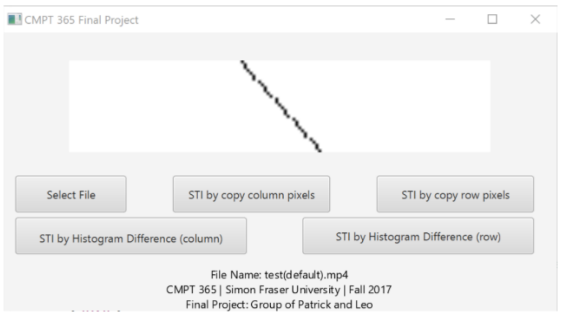

CMPT 365 Final Project
Select File
STI by copy column pixels
STI by copy row pixels
STI by Histogram Difference (column)
STI by Histogram Difference (row)
File Name: test(default).mp4
CMPT 365 | Simon Fraser University | Fall 2017
Final Project: Group of Patrick and Leo

Fig. 3(a)

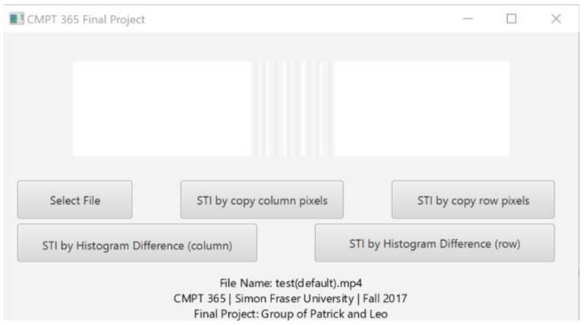

CMPT 365 Final Project
Select File
STI by copy column pixels
STI by copy row pixels
STI by Histogram Difference (column)
STI by Histogram Difference (row)
File Name: test(default).mp4
CMPT 365 | Simon Fraser University | Fall 2017
Final Project: Group of Patrick and Leo

Fig. 3(b)

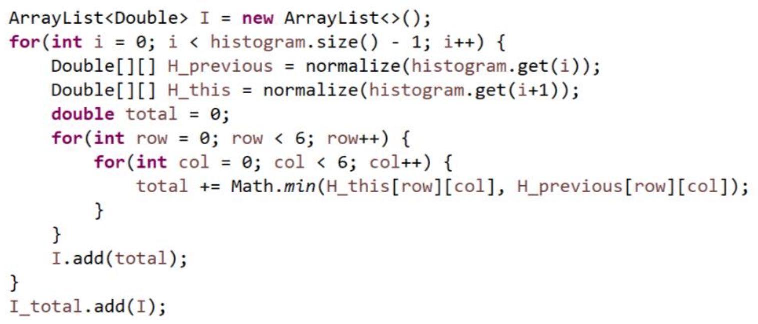
If you select “STI by Histogram Difference (column)” button, the program will immediately display the oblique line, shown in Fig. 3(a). When you click “STI by Histogram Difference (row)” button, they will give you image like Fig. 3(b).

Things can be improved:

Due to the time issue, we still have some ideas for this project that have not implemented yet.

- Firstly, according to the project instruction, we could add an extra button to set the value of threshold for the histogram differences. The users could obtain anticipated images through the customized threshold value.

- Furthermore, we also could optimize our algorithm for calculating the histogram intersections. As the instruction mentioned, we could set a value z that denotes the difference between histograms. To calculate the squared Euclidean distance, we could use the product of z and the transpose of z with the matrix A, which is the matrix to declare the nearness of the colors.



```
ArrayList<Double> I = new ArrayList<>();
for(int i = 0; i < histogram.size() - 1; i++) {
    Double[][] H_previous = normalize(histogram.get(i));
    Double[][] H_this = normalize(histogram.get(i+1));
    double total = 0;
    for(int row = 0; row < 6; row++) {
        for(int col = 0; col < 6; col++) {
            total += Math.min(H_this[row][col], H_previous[row][col]);
        }
    }
    I.add(total);
}
I_total.add(I);
```

Fig.4: Code segment for histogram difference.

Features we implemented:

- Fast processing due to resizing the image, we can get the final displayed image faster without losing details.

- Implementing a backup frame buffer to avoid user clicking the buttons without order to cause runtime error. E.g.: try to display STI image without selecting videos
- A video file name display text field that indicates the name of the file user selected.

Conclusion:

From this project, we mastered the techniques and knowledge for applying the OpenCV library, JavaFX, and other tools to implement digital multimedia programs. In this project, we have implemented the application for constructing the STI by copying the center column or row pixels and the histogram difference in each column or row. Even though there are many functions that still could be improved, our application has followed the project instruction and has met all the requirements.