**CS4187 Computer Vision & Interactivity**

Assignment 2 Report

Name: Jaya Ramesh CHALIKI

Student ID: 55360451

Email: [jrchaliki2-c@my.cityu.edu.hk](mailto:jrchaliki2-c@my.cityu.edu.hk)

Setup:

I have included only src folder for each question. Simply replace them with the relevant project. If any problems are encountered, please contact. Done on Xcode.

***Question 1:***

For this question, we setup variables for loading videos and converting each frame to matrices. We also define two sliders for low threshold for edge detection and line vote threshold for Hough Transform. Load the video into vid and play is using vid.play().

In update function, we check for new frames and extract pixels from each frame.

The pipeline has the following steps:

1. In each new frame in update(), apply gaussian filter to smooth out noise.
2. Apply Canny Edge Detection per low threshold.
3. Apply Hough Transform Lines as per vote threshold

Whenever user clicks on the area of video, we extract the co-ordinates and save them, when 3 or more points are clicked, we form a polygon mask as a region of interest. The code is carried out as taught in tutorial session.

Screenshot:

A screenshot of a computer screen

Description automatically generated

***Question 2:***

We first use videograbber class, set height and width for our video. Declare variables for img , mat and histogram.

Procedure:

1. It is similar to Q01 but here we use Hough Circles function to detect circles for us.
2. Declare variables for webcam and relevant matrices we need for our operations.
3. Check for frame changes in update function.
4. Apply gaussian filter on each frame, canny edge function and finally Hough Circle Transforms.
5. The output circles provides us the location to draw black spot wherever we detect a Circle.

Results:

A picture containing person, camera, mirror, photo

Description automatically generated

Question 3:

For this question, we setup variables for loading images and converting to matrices. We also define a slider for low threshold for edge detection.

The pipeline has the following steps:

1. In each image in update(), apply gaussian filter to smooth out noise.
2. Apply Canny Edge Detection per low threshold.
3. We reset timer in setup
4. Declare points from Point to store the edge information we need for drawing circles.
5. Once edge is detected, we run two for loops to store the points and ofDrawCircle draws circle and moves along the points of edges.
6. We have Booleans like notfinished to give that coming back effect.
7. Update checks for threshold changes for canny edge detection.

Results:

A picture containing graphical user interface

Description automatically generated