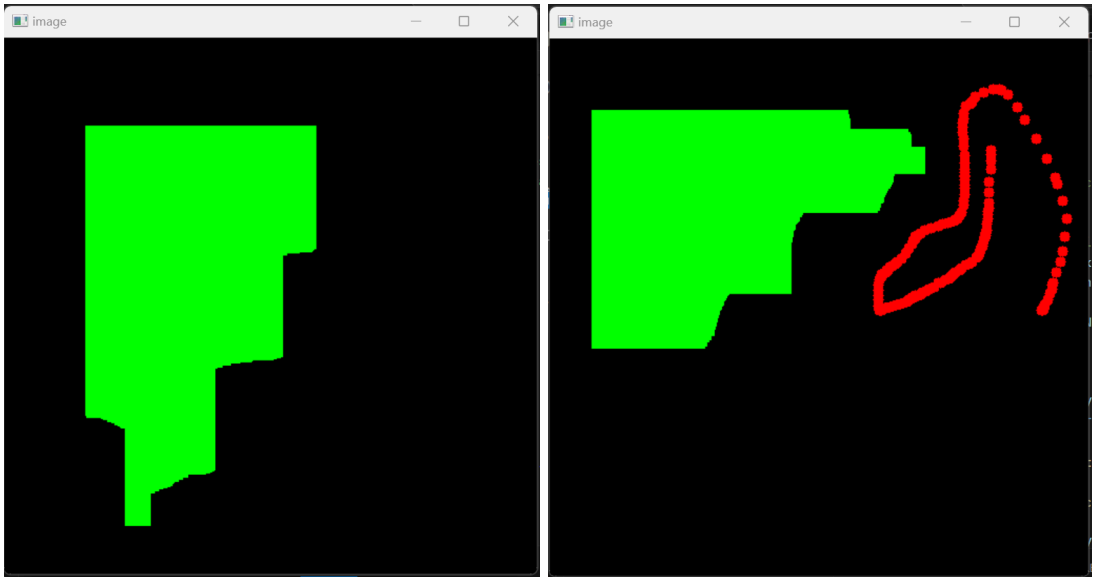
COS 470 Image Processing and Computer Vision
2024 Fall
Taylor Brookes
09/23/2024
Assignment 1

Q1.1: Write a summary of what you learned from the tutorial "Mouse as a Paint-Brush", https://docs.opencv.org/4.10.0/db/d5b/tutorial_py_mouse_handling.html
A1.1:

The first example code showed all the available events that can be used.

The second example code begins by creating a black (3) 512x512 pixel window which, when double-clicked (event), creates a filled (-1) blue (255,0,0) circle with radius of 100 pixels centered where the mouse is double-clicked in the window. When the escape key (== 27) is pressed, the window closes. The circle could be an outline instead of filled in if the -1 is changed to a positive value.

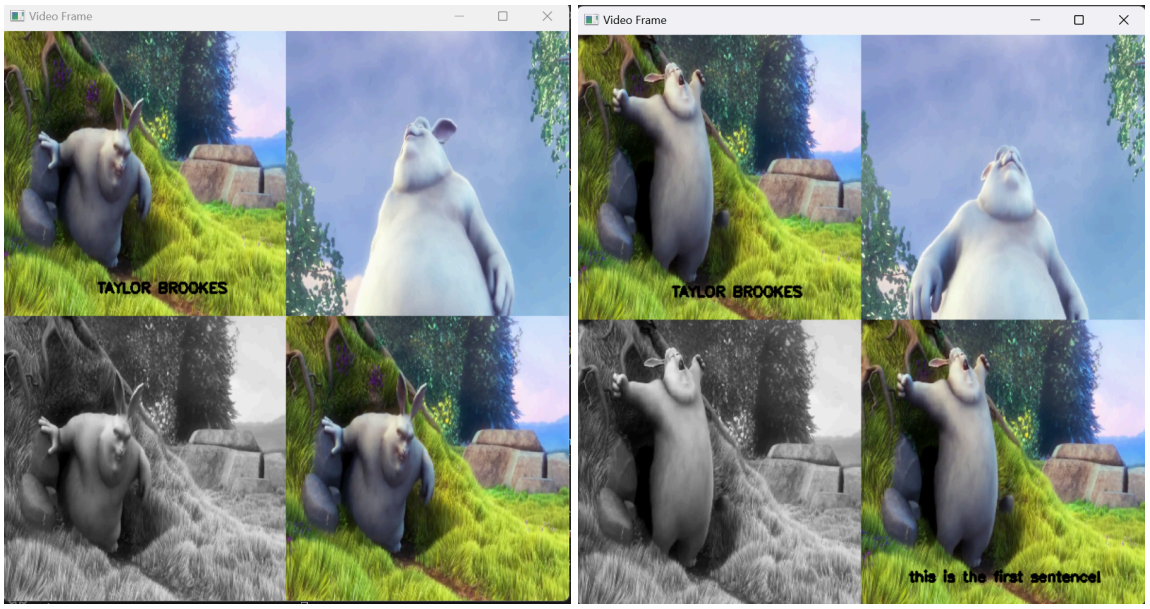
The third (and fourth combined) example code begins with the same black image, and then creates a filled green rectangle starting at the initial click and ending where the mouse is dragged to and released by default. When “m” is pressed, the operation changes from a rectangle with boundaries set by the mouse to a line of red circles with radius of 5 pixels. “m” can be changed on line 37 with the code: ord(‘m’) to be any character we choose. As with before, the filled circles and rectangles can be changed to an outline instead by changing the -1 parameter to a positive value. If we change the initial “mode = True” code on line 5 to “False”, the program begins with drawing a line of circles and then pressing “m” changes it to draw a rectangle. Interestingly, a phenomenon I witnessed is that the rectangle is drawn progressively, meaning that if you go in one X or Y direction and then change the direction while still drawing to the point of repassing the origin, instead of it changing the entire rectangle from the start in most programs, it just continues drawing the rectangle over itself, as shown in the image below.

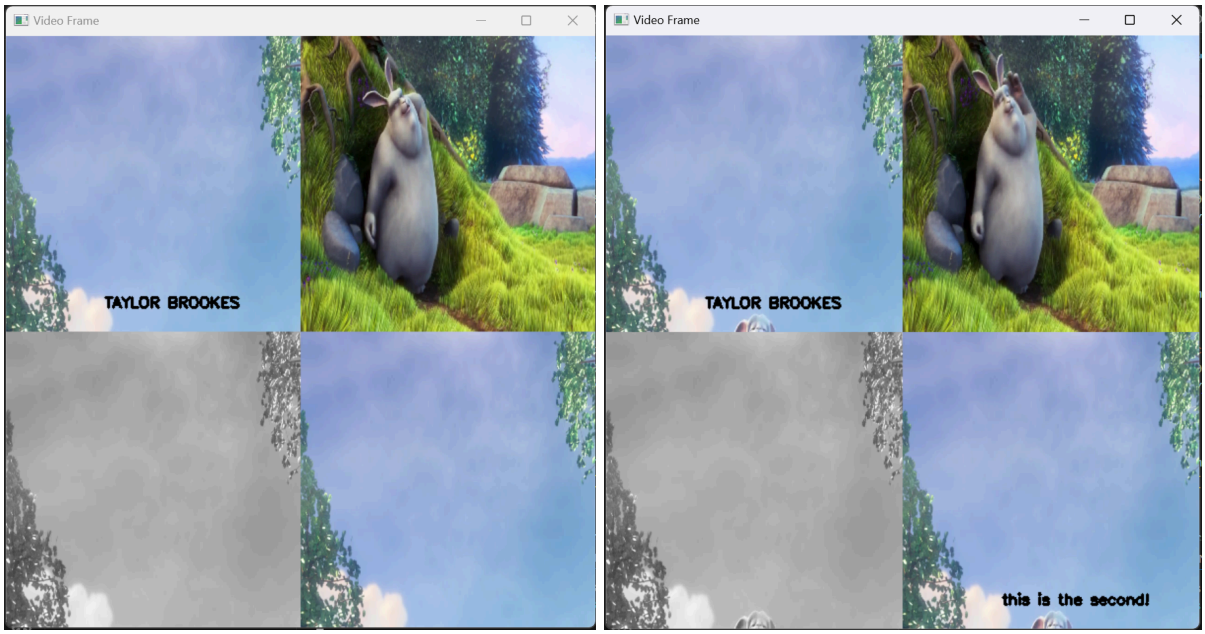


Q1.2: Navigate to “cv::Sobel()” and describe the information presented on the documentation page and explain how the content helps in effectively understanding the use of the function.
A1.2: Just like how we learned to use Javadoc in the intro to Java courses, this OpenCV cv::Sobel() page describes how the function cv.Sobel() is used. The type of the parameter as well as each parameter’s meaning and use is laid out. In addition, a short explanation of what the Sobel operator does and how to effectively use it is described.

Q2: Make a 2x2 grid of the same video 4 times with the following edits:
YOUR_NAME watermark
reverse playback
grayscale
subtitles from “subtitle.txt”

A2:






Video Frame
TAYLOR BROOKES

Q3: Modify and save the video file with the following edits:
2x speed
4x speed
0.5x speed
A3: I used an approach of increasing the framerate of the .avi output video for 2x and 4x faster, and kept the original framerate of 25FPS but wrote each frame twice to achieve 0.5x speed.

Alternatively, I messed with time.perf_counter_ns() via this link (https://stackoverflow.com/questions/5998245/how-do-i-get-the-current-time-in-milliseconds-in-python) and created this table with one time record right before the main while loop and another in before the break statement. The values do not directly line up with what is asked for, so I chose the other approach because it is less dependent upon the hardware of the user running the program. This is also dependent upon displaying the video while running the program, instead of the video just being saved as requested.

.waitKey(MS_DELAY_HERE)	duration of video
1	1.338sec
2	4.570sec
5	4.858sec
10	5.773sec
11	5.607sec
12	5.486sec
13	7.721sec
20	10.517sec
25	10.480sec
27	10.627sec
28	13.984sec
29	14.289sec
30	15.158sec
50	21.057sec
70	26.395sce
72	26.683sec