

Real-time Canny edge detection with a webcam

Orion Peter Fernandes - 65047854

1. Project Description (2 pts)

- This project deals with canny edge detection using a webcam implemented through a Raspberry Pi 4.
- With the sample code taken as example to modify a single frame, it has to be modified to process multiple frames in real time.
- With the provided OpenMP and pthread files, they have to be modified similarly
- The FPS outputs from each file execution is measured and parallelization is added to *canny_util.c/.h*
- I would expect a faster FPS with OpenMP and pthreads as compared to the original.

2. Experimental Setup (4 pts)

- Download the provided *canny_util.c*, *canny_util.hand* and *camera_canny.cpp* from Canvas and transfer it onto the Raspberry Pi.
- Ensure camera module is plugged into the board into the right manner.

- After powering Pi, make sure system is upto date with

\$ sudo apt-get update && sudo apt-get upgrade

- Ensure raspberry pi camera module is enabled:

\$ sudo raspi-config

- Take a sample image and video to make sure the camera runs smoothly:

\$ raspistill -o demo.jpg

\$ raspivid -o demo.h264 -t 10000

- To get the raspicam library to control the module over the C++ code, use the following code to get and install the necessary files into raspicam directory:

```
$ git clone https://github.com/cedricve/raspicam ./raspicam
```

```
$ cd raspicam
```

```
$ mkdir build
```

```
$ cd build
```

```
$ sudo apt-get install cmake
```

```
$ cmake ..
```

```
$ make
```

```
$ sudo make install
```

```
$ sudo ldconfig
```

```
$sudo apt-get install libopencv-dev
```

- Check the success of installation of this module by the command:

```
$raspicam_test
```

- Proceed to check the OpenCV version with:

```
pkg-config --modversion opencv
```

- Include OpenCV libraries among header files in code.
- Compile the downloaded source code use the following command:

```
$ g++ `pkg-config --cflags --libs opencv` -I. canny_util.c camera_canny.cpp -o  
camera_canny
```

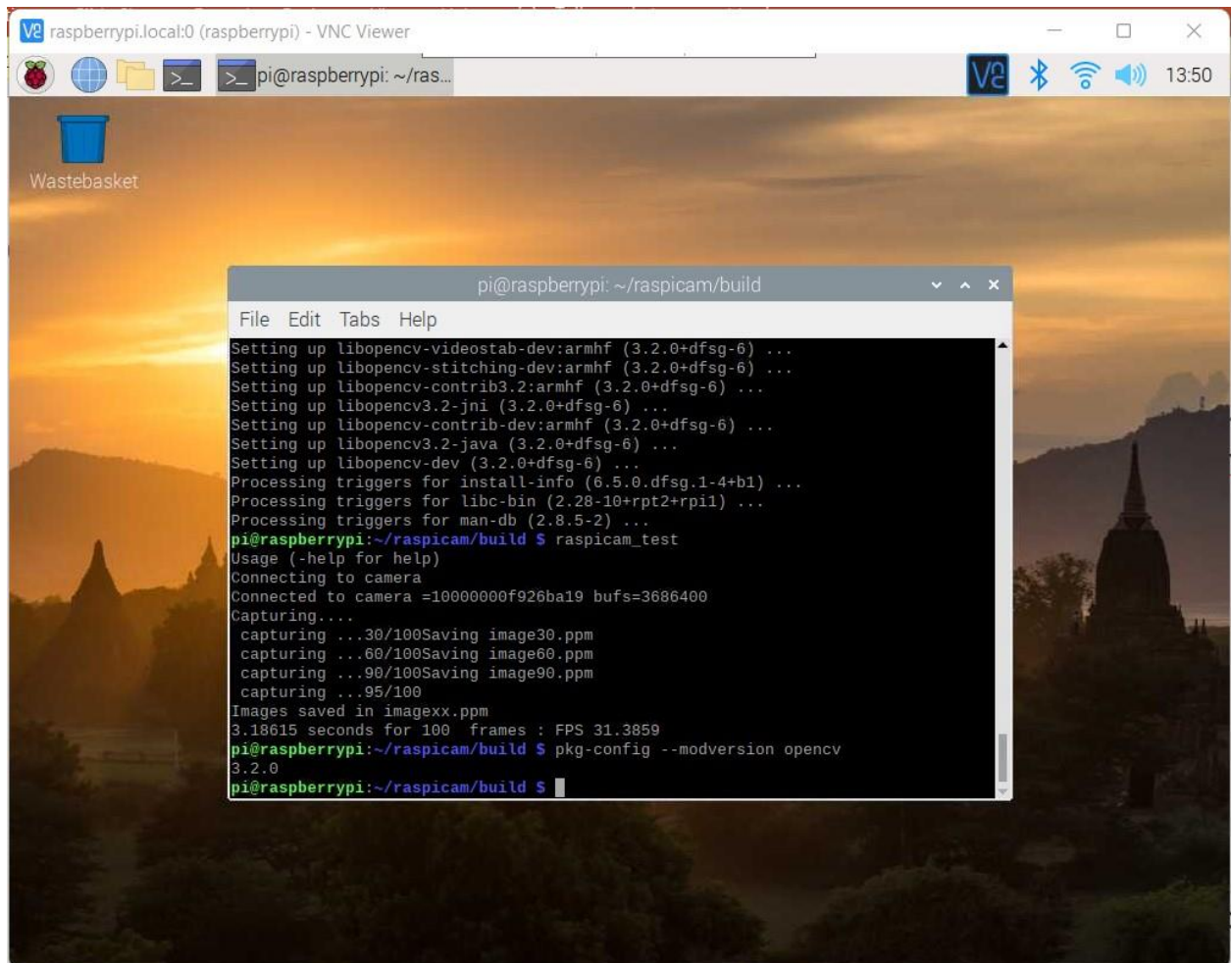
- To test, type:

```
$ ./camera_canny 1.0 0.2 0.6
```

- Extend code for processing multiple images such that it saves processed images numerically.
- Add the FPS calculator that averages values of timing for multiple images.
- Parallelize the *canny_util.c/.h* code by splitting functions to run concurrently.

- Apply multi-threading implementation of pthreads and OpenMP from Assignment2 to *canny_util.c/.h*
- Add the name of the flags while compiling the code (*-lpthreads, -fopenmp*)

3. Results (6 pts)



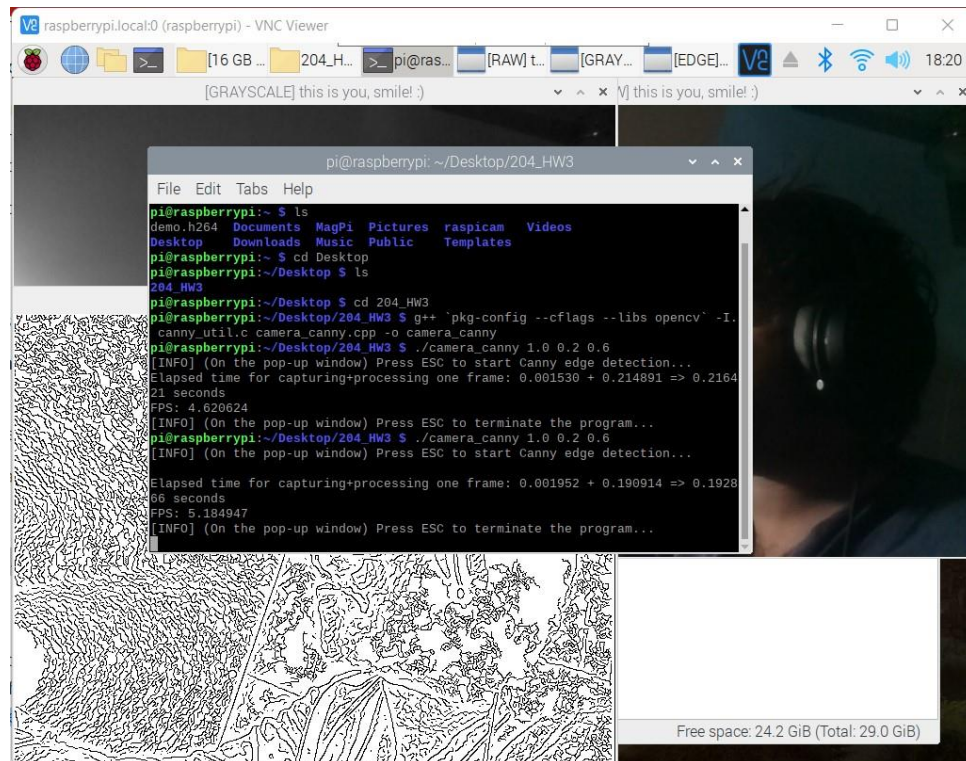
The screenshot shows a VNC Viewer window titled 'raspberrypi.local:0 (raspberrypi)'. The desktop background is a sunset over a landscape with a pagoda. A terminal window titled 'pi@raspberrypi: ~/raspicam/build' is open, displaying the following output:

```

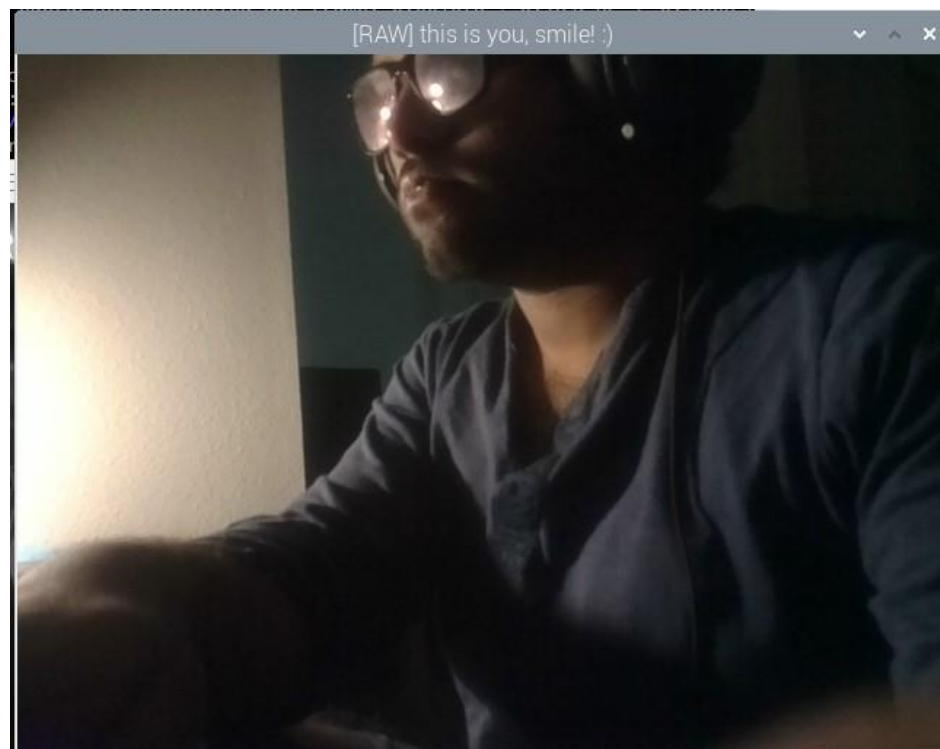
File Edit Tabs Help
Setting up libopencv-videostab-dev:armhf (3.2.0+dfsg-6) ...
Setting up libopencv-stitching-dev:armhf (3.2.0+dfsg-6) ...
Setting up libopencv-contrib3.2:armhf (3.2.0+dfsg-6) ...
Setting up libopencv3.2-jni (3.2.0+dfsg-6) ...
Setting up libopencv-contrib-dev:armhf (3.2.0+dfsg-6) ...
Setting up libopencv3.2-java (3.2.0+dfsg-6) ...
Setting up libopencv-dev (3.2.0+dfsg-6) ...
Processing triggers for install-info (6.5.0.dfsg.1-4+b1) ...
Processing triggers for libc-bin (2.28-10+rpt2+rpil) ...
Processing triggers for man-db (2.8.5-2) ...
pi@raspberrypi:~/raspicam/build $ raspicam_test
Usage (-help for help)
Connecting to camera
Connected to camera =10000000f926ba19 bufs=3686400
Capturing....
capturing ...30/100Saving image30.ppm
capturing ...60/100Saving image60.ppm
capturing ...90/100Saving image90.ppm
capturing ...95/100
Images saved in imagexx.ppm
3.18615 seconds for 100 frames : FPS 31.3859
pi@raspberrypi:~/raspicam/build $ pkg-config --modversion opencv
3.2.0
pi@raspberrypi:~/raspicam/build $

```

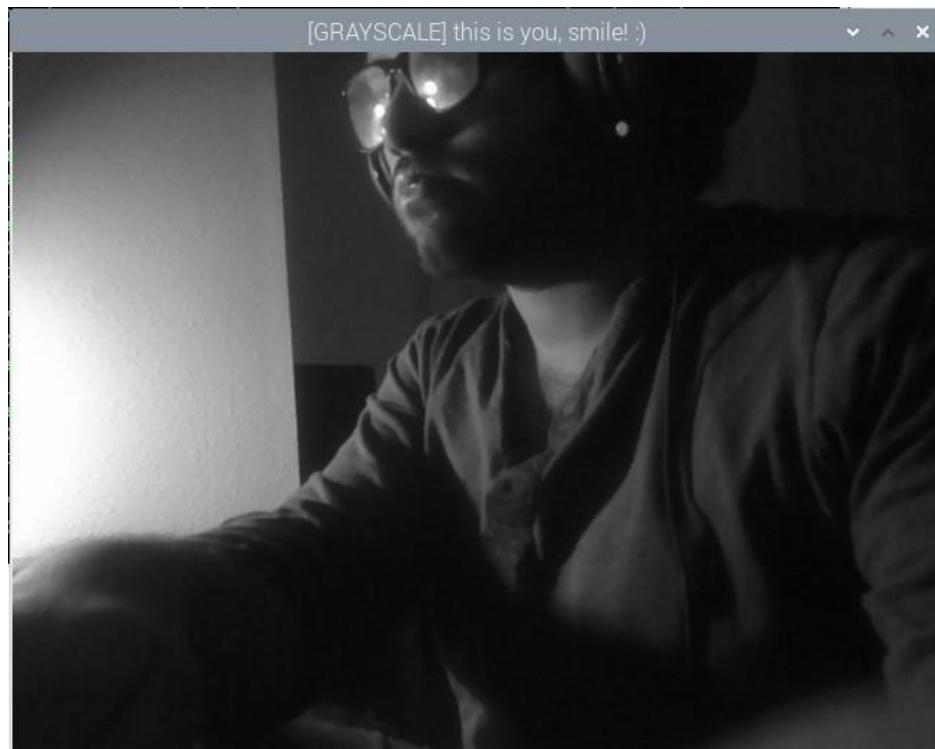
Installing metapackage for development to OpenCV and checking version



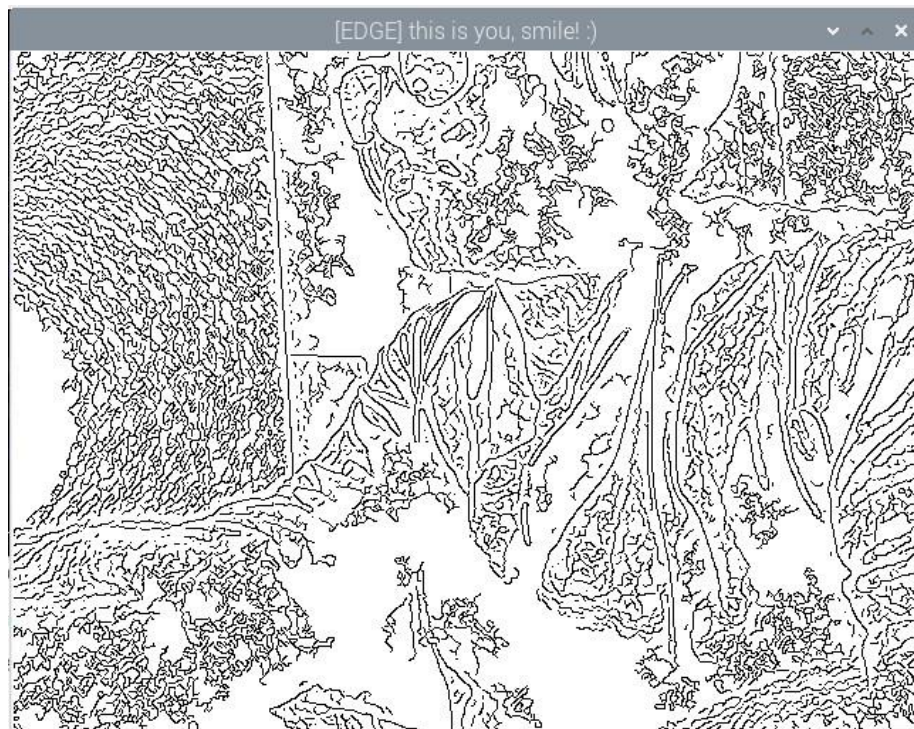
Compiling code for canny_util.c



Output Image (canny_util.c)



Grayscale image (canny_util.c)



Edge output (canny_util.c)

```
pi@raspberrypi: ~/Desktop/204_HW3
File Edit Tabs Help

canny_local_omp.c: In function 'int non_max_supp(short int*, short int*, short i
nt*, int, int, unsigned char)':
canny_local_omp.c:891:1: warning: no return statement in function returning non-
void [-Wreturn-type]
}
^
pi@raspberrypi:~/Desktop/204_HW3 $ ./camera_canny 1.0 0.2 0.6
[INFO] (On the pop-up window) Press ESC to start Canny edge detection...

Elapsed time for capturing+processing one frame: 0.001675 + 0.184249 => 0.185924
seconds
FPS: 5.378542
[INFO] (On the pop-up window) Press ESC to terminate the program...
^pi@raspberrypi:~/Desktop/204_HW3 $
pi@raspberrypi:~/Desktop/204_HW3 $ g++ `pkg-config --cflags --libs opencv` -I. c
anny_util.c camera_canny.cpp -o camera_canny
pi@raspberrypi:~/Desktop/204_HW3 $ ./camera_canny 1.0 0.2 0.6
[INFO] (On the pop-up window) Press ESC to start Canny edge detection...
Elapsed time for capturing+processing one frame: 0.001557 + 0.180949 => 0.182506
seconds
FPS: 5.479272
[INFO] (On the pop-up window) Press ESC to terminate the program...
```

FPS Output (canny_util.c) – 5.479

- Parallelized Blur Y function (pthread) –

```
void *blur_y(void *arguments)
{
/*****
* Blur in the y - direction.
*****/
struct thread_args_y *args = arguments;

unsigned char *image = args->image;
int rows = args->rows;
int cols = args->cols;
int row_s = args->row_s;
int row_e = args->row_e;
int center = args->center;
float *kernel = args->kernel;
float *tempim = args->tempim;

int r, c, rr;
float dot, sum;

if(VERBOSE) printf("    Blurring the image in the Y-direction.\n");
```

```

        for(c=0;c<cols;c++){
            for(r=row_s;r<row_e;r++){
                dot = 0.0;
                sum = 0.0;
                for(rr=(-center);rr<=center;rr++){
                    if(((c+cc) >= 0) && ((r+rr) < rows)){
                        dot += (float)image[(r+rr)*cols+c] *
kernel[center+rr];
                        sum += kernel[center+rr];
                    }
                }
                tempim[r*cols+c] = (short int) (dot*BOOSTBLURFACTOR/sum +
0.5);
            }
        }
    }
}

```

- Parallelized Blur Y function (OpenMP) -

```

/*****
 * Blur in the y - direction.
*****/

int p_thread = 4;
omp_set_dynamic(0);
omp_set_num_threads(p_thread);
#pragma omp parallel private(r, rr, dot, sum)
{
    if (VERBOSE) printf("    Blurring the image in the Y-direction.\n");
    for (c = 0; c < cols; c++) {
        for (r = 0; r < rows; r++) {
            sum = 0.0;
            dot = 0.0;
            for (rr = (-center); rr <= center; rr++) {
                if ((r + rr) >= 0 && (r + rr) < rows) {
                    dot += (float) image[(r + rr) * cols + c] *
kernel[center + rr];
                    sum += kernel[center + rr];
                }
            }
            twmpim[r * cols + c] = (short int) (dot * BOOSTBLURFACTOR /
sum + 0.5);
        }
    }
}

```

```

    }
    }
    free(tempim);
    free(kernel);
}
}

```

- While loop to capture 1000 frames -

```

begin = clock();
while(int n=0, n<NFRAME){

    //capture
    cap >> frame;
    mid = clock();
    cvtColor(frame, grayframe, CV_BGR2GRAY);
    image = grayframe.data;
}

```

- FPS modified to average over 100 frames -

```

printf("FPS: %01lf\n", time_elapsed/NFRAME);

```

- File creation added within loop to create 100 images –

```

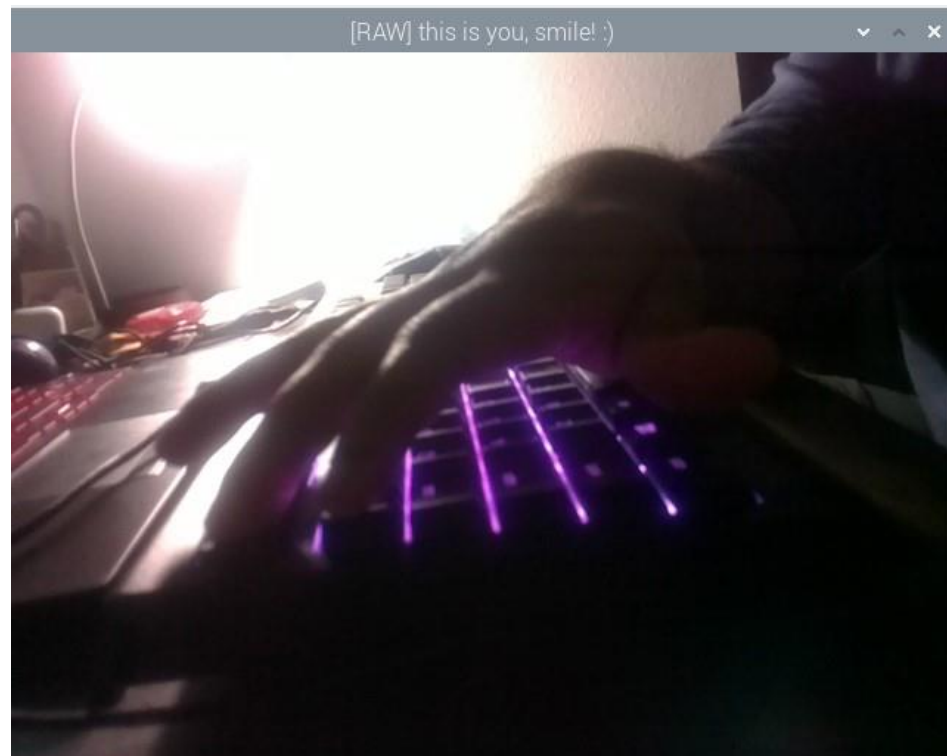
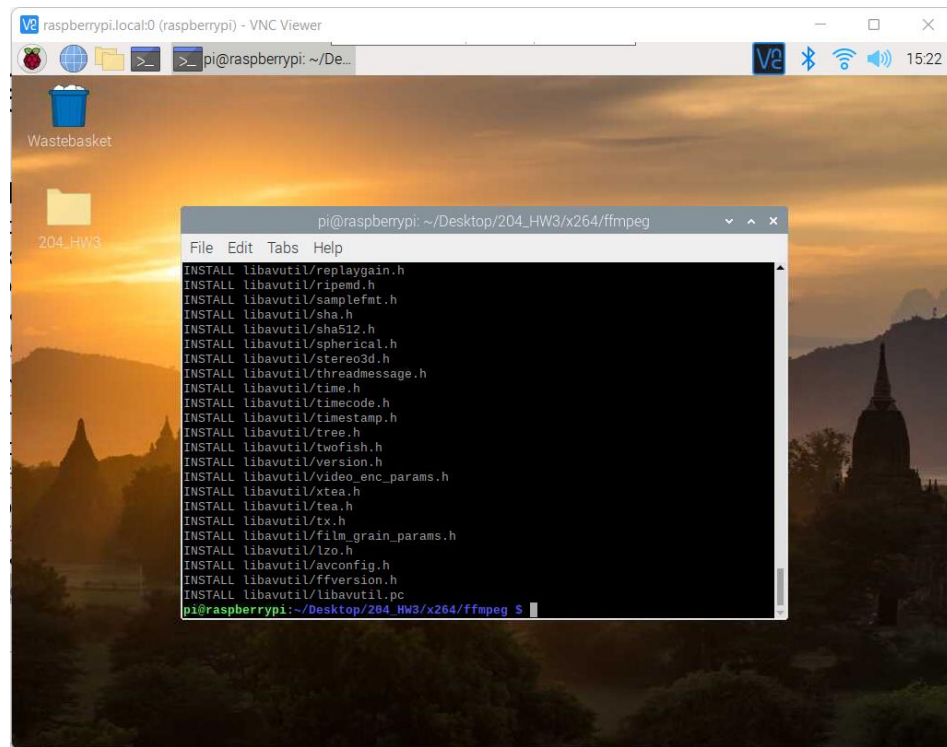
if(VERBOSE) printf("Starting Canny edge detection.\n");
if(dirfilename != NULL){
    sprintf(composedfname, "Frame%n.fim",
        sigma, tlow, thigh);
    dirfilename = composedfname;
}
canny(image, rows, cols, sigma, tlow, thigh, &edge, dirfilename);

/*****
* Write out the edge image to a file.
*****/
while(int n=0, n<NFRAME, n++){
    {
        sprintf(outfilename, "Frame%n.pgm", sigma, tlow, thigh);
    }

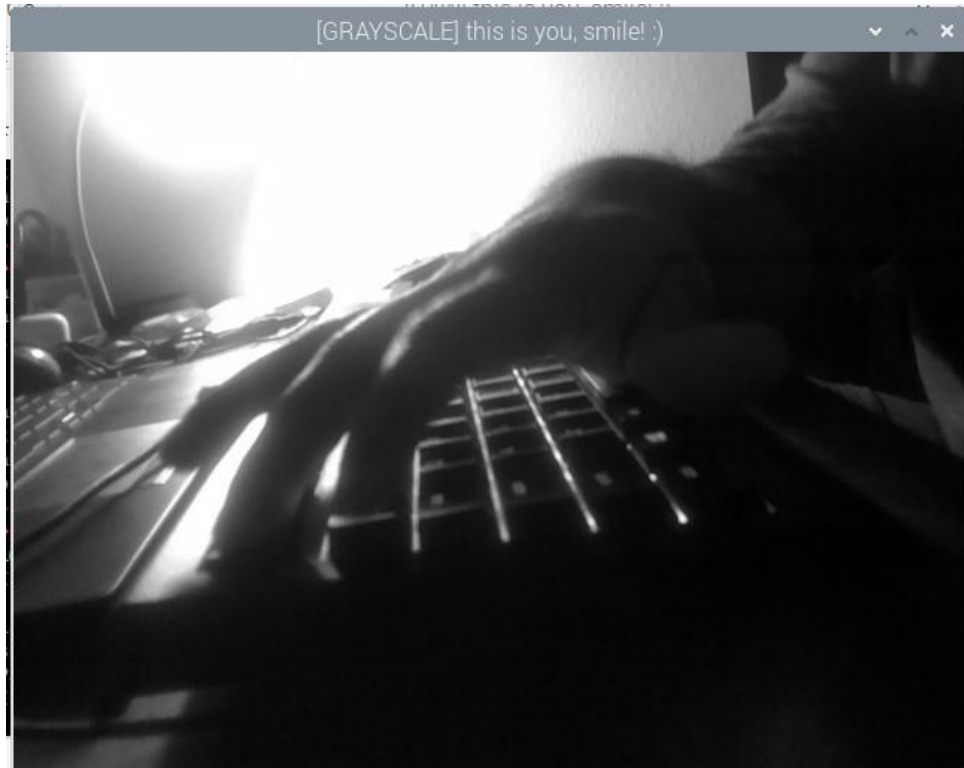
    if (VERBOSE) printf("Writing the edge image in the file %s.\n", outfile);
    if (write_pgm_image(outfilename, edge, rows, cols, NULL, 255) == 0) {
        fprintf(stderr, "Error writing the edge image, %s.\n", outfile);
        exit(1);
    }
}

```

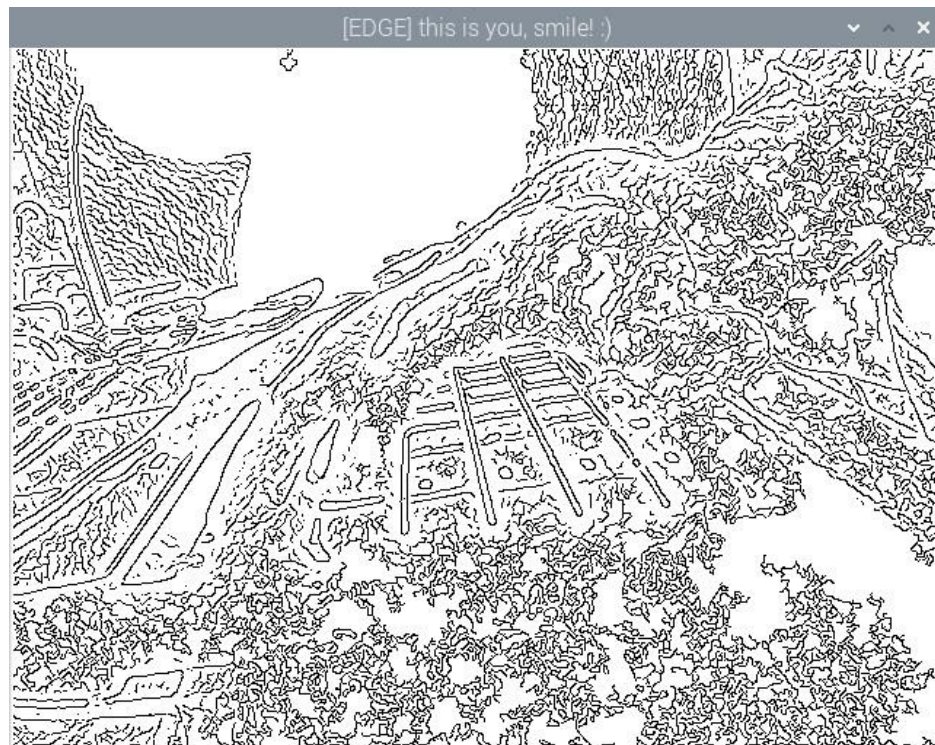

- Installing ffmpeg codec –



Output Image (OpenMP)



Grayscale Image (OpenMP)

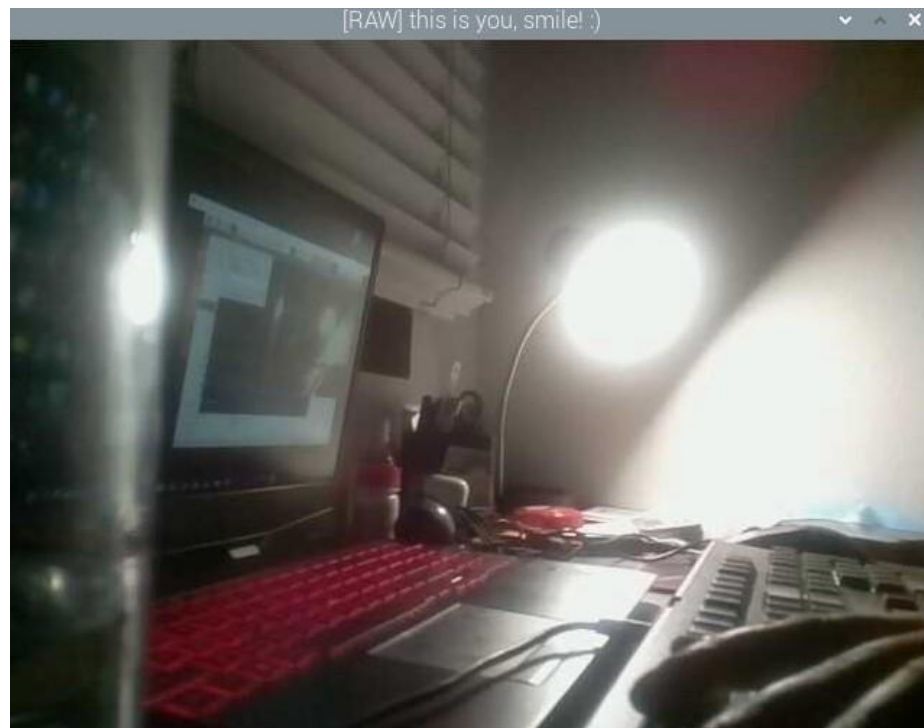


Edge Output (OpenMP)

```
pi@raspberrypi: ~/Desktop/204_HW3
File Edit Tabs Help
rt int, int)':
canny_local_omp.c:578:1: warning: no return statement in function returning non-void [-Wreturn-type]
}
^
canny_local_omp.c: At global scope:
canny_local_omp.c:695:26: error: ISO C++ forbids declaration of 'non_max_supp' with no type [-fpermissive]
    unsigned char *result)
    ^
canny_local_omp.c: In function 'int non_max_supp(short int*, short int*, short int*, int, int, unsigned char*)':
canny_local_omp.c:891:1: warning: no return statement in function returning non-void [-Wreturn-type]
}
^
pi@raspberrypi:~/Desktop/204_HW3 $ ./camera_canny 1.0 0.2 0.6
[INFO] (On the pop-up window) Press ESC to start Canny edge detection...

Elapsed time for capturing+processing one frame: 0.001675 + 0.184249 => 0.185924 seconds
FPS: 5.378542
[INFO] (On the pop-up window) Press ESC to terminate the program...
```

FPS Output (OpenMP) – 5.378542



Output Image (pthreads)



Grayscale Image (pthreads)



Edge Output (pthreads)

```
pi@raspberrypi: ~/Desktop/204_HW3
File Edit Tabs Help
canny_local_pthread.c: In function 'int follow_edges(unsigned char*, short int*,
short int, int)':
canny_local_pthread.c:630:1: warning: no return statement in function returning
non-void [-Wreturn-type]
}
^
canny_local_pthread.c: At global scope:
canny_local_pthread.c:747:26: error: ISO C++ forbids declaration of 'non_max_sup
p' with no type [-fpermissive]
    unsigned char *result)
    ^
canny_local_pthread.c: In function 'int non_max_supp(short int*, short int*, sho
rt int*, int, int, unsigned char*)':
canny_local_pthread.c:943:1: warning: no return statement in function returning
non-void [-Wreturn-type]
}
^
pi@raspberrypi:~/Desktop/204_HW3 $ ./camera_canny 1.0 0.2 0.6
[INFO] (On the pop-up window) Press ESC to start Canny edge detection...
Elapsed time for capturing+processing one frame: 0.001820 + 0.199921 => 0.201741
seconds
FPS: 4.956851
[INFO] (On the pop-up window) Press ESC to terminate the program...
pi@raspberrypi:~/Desktop/204_HW3 $
```

FPS Output (pthreads) – 4.956851

Output Table

Method	FPS Output
No Parallelization	5.479
Parallelization with OpenMP	5.378542
Parallelization with pthreads	4.956851

- Considering the FPS outputs from the different methods employed during execution yielded different results.
- The OpenMP FPS output was similar to the normal execution of canny_util.c with a slight difference of 0.1.
- pthreads had the best throughput among the rest with 4.956 units.

4. Problems and Discussion (6 pts)

- canny_util.c, canny_local_omp.c, canny_local_pthread.c have missing header files to utilize OpenCV while execution.
- Solution – added header files individually to each file - #include “opencv2/opencv.hpp”
- OpenCV package not found even after adding header files.
- Solution in reference to <https://stackoverflow.com/questions/15320267/package-opencv-was-not-found-in-the-pkg-config-search-path> - made a file named "opencv.pc" and copied it to "/usr/local/lib/pkgconfig" Then i added these two lines to ".bashrc":

```
PKG_CONFIG_PATH=$PKG_CONFIG_PATH:/usr/local/lib/pkgconfig  
export PKG_CONFIG_PATH
```

- Compilation of canny_util.c, canny_local_omp.c, canny_local_pthread.c produce warnings when compiling.
- The canny edge program fails to terminate even after pressing escape (ESC) since it doesn't accept output when not on current window.
- The OpenMP and pthread files could have OpenCV header files pre-added for simpler execution.
- Current Raspbian images don't support raspicam which makes performing the project exceptionally hard. Gives error even after updating and upgrading–

```
-bash: raspistill: command not found
```

- Solution: Had to downgrade from Debian to buster which took a considerable amount of time and resources.
- The outputs vary a bit with pthreads producing a faster output but didn't produce quite an edge image as compared to the others.
- Solution: A single image can be used to test out different execution methods to attain accurate FPS results.
- As expected, parallelization with pthreads had the best throughput due to dedicated thread processing, reducing the time taken to process the image through canny detector.
- While modifying camera_canny, several variables had to be modified and declared to run error-free.
- Parts of code in camera_canny omitted to reduce memory usage and improve output.

5. Conclusion (2 pts)

- We can conclude from this project that the edge detector could still undergo more improvements and stages to attain a better efficiency and throughput for real time applications.
- For a considerable amount of 1000 frames in this project, the edge detector could be implemented in real time system if needed but would require more optimization given the current output.

6. References

- clock() - <https://stackoverflow.com/questions/2962785/c-using-clock-to-measure-time-in-multi-threaded-programs>
- OpenCV - <https://docs.opencv.org/5.x/d1/dfb/intro.html>
- OpenCV Error – <https://stackoverflow.com/questions/15320267/package-opencv-was-not-found-in-the-pkg-config-search-path>
- Pthread.h - <https://askubuntu.com/questions/420722/how-to-compile-a-c-program-that-uses-pthread-h>
- -lm -fopenmp -pthread flag error - <https://stackoverflow.com/questions/3949161/no-such-file-or-directory-but-it-existslow>