July 21, 2017

ECEN 5623 – Real Time Embedded Systems

Summer 2017

EXERCISE 4

Real Time Continuous Media

Jeet Baru

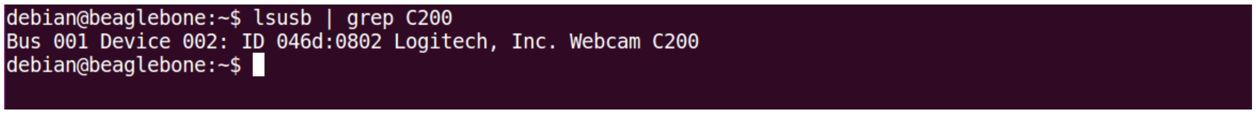
Email: jeet.baru@colorado.edu

ID Number: 107189037

Question 1

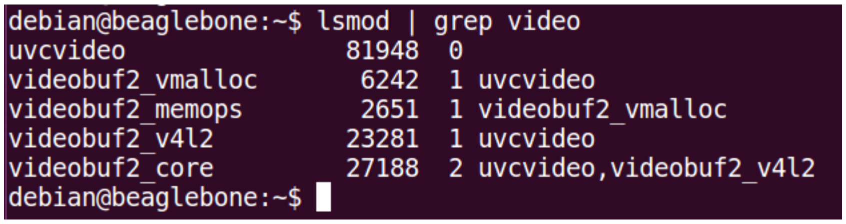
In this question we are required to connect our Logitech C200 camera tour BeagleBone Board and verify whether it is being detected. For this we use lsusb, lsmod and dmesg commands.

On connecting our camera via USB to our BeagleBone we verify its detection via the lsusb command. This command displays the usb busses in the system and devices connected to them. On running the command, we get the following output depicting correct detection of C200.



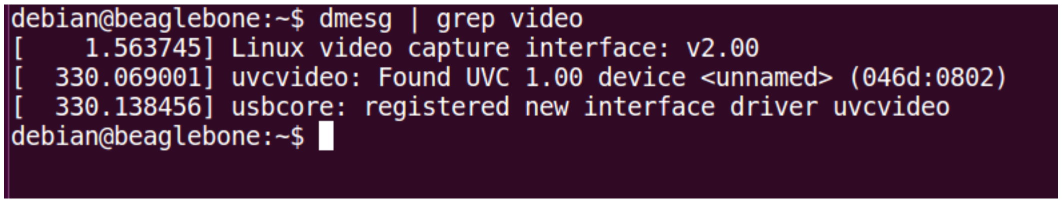
```
debian@beaglebone:~$ lsusb | grep C200
Bus 001 Device 002: ID 046d:0802 Logitech, Inc. Webcam C200
debian@beaglebone:~$
```

Next, we run the lsmod command to ensure that UVC driver is loaded as well. Lsmod command shows which loadable kernel modules are currently loaded. On running the command, we observe that the uvc video driver is a loaded module and hence the camera is ready to use.



```
debian@beaglebone:~$ lsmod | grep video
uvcvideo               81948  0
videobuf2_vmalloc       6242  1 uvcvideo
videobuf2_memops        2651  1 videobuf2_vmalloc
videobuf2_v4l2         23281  1 uvcvideo
videobuf2_core         27188  2 uvcvideo,videobuf2_v4l2
debian@beaglebone:~$
```

Now we run the dmesg command to verify that the uvcdriver was loaded. On running the command, we see a timestamp and the kernel message for the same.



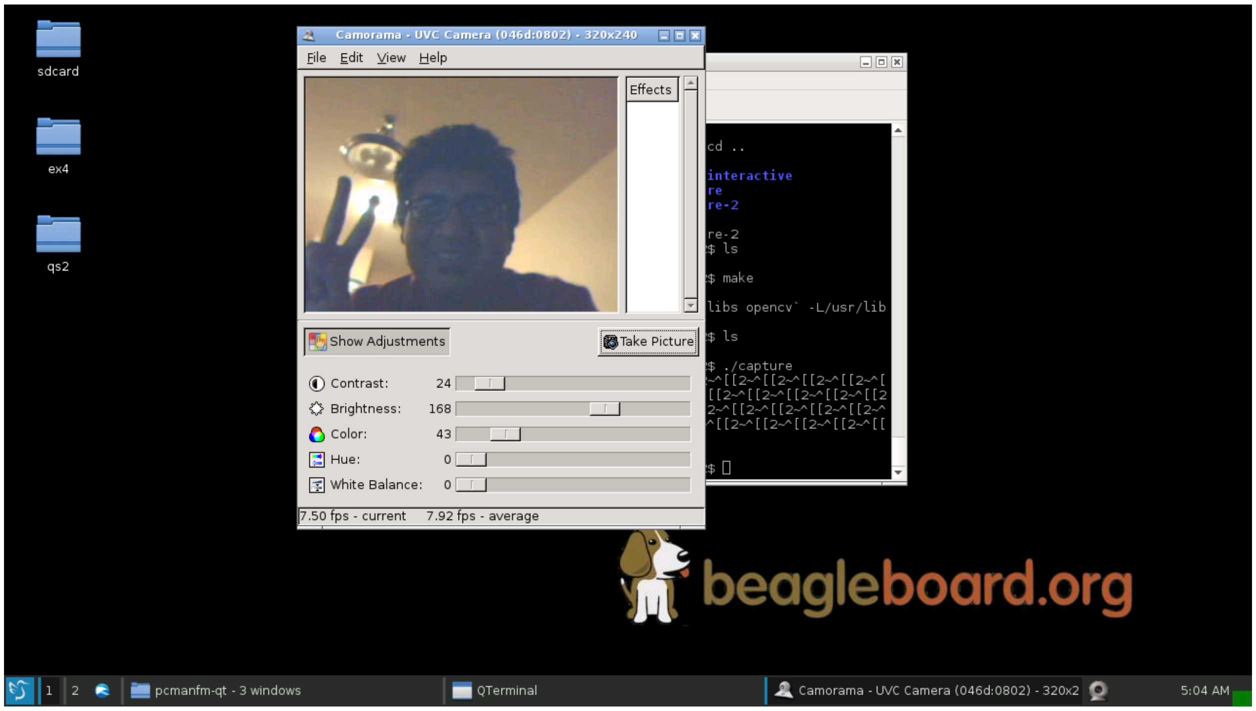
```
debian@beaglebone:~$ dmesg | grep video
[    1.563745] Linux video capture interface: v2.00
[  330.069001] uvcvideo: Found UVC 1.00 device <unnamed> (046d:0802)
[  330.138456] usbcore: registered new interface driver uvcvideo
debian@beaglebone:~$
```

UVC driver is a USB Video Class driver that provides driver support for USB Video Class devices i.e., it enables the video streaming functionality on the USB bus.

Hence, we have our camera ready to stream media.

Question 2

In this question we install camorama, it provides nice camera capture GUI tools. On running the application, we can change the contrast, hue, white balance, brightness and color. Screenshot depicting installation and verifying the functioning of camera is shown below:



```
Contrast:       24
Brightness:    168
Color:          43
Hue:             0
White Balance:   0
7.50 fps - current    7.92 fps - average
```

```
cd ..

interactive
re
re-2

re-2
$ ls

$ make

libs opencv` -L/usr/lib

$ ls

$ ./capture
~^[[2~^[[2~^[[2~^[[2~^[
[[2~^[[2~^[[2~^[[2~^[[2
2~^[[2~^[[2~^[[2~^[[2~^
^[[2~^[[2~^[[2~^[[2~^[[


$ 
```

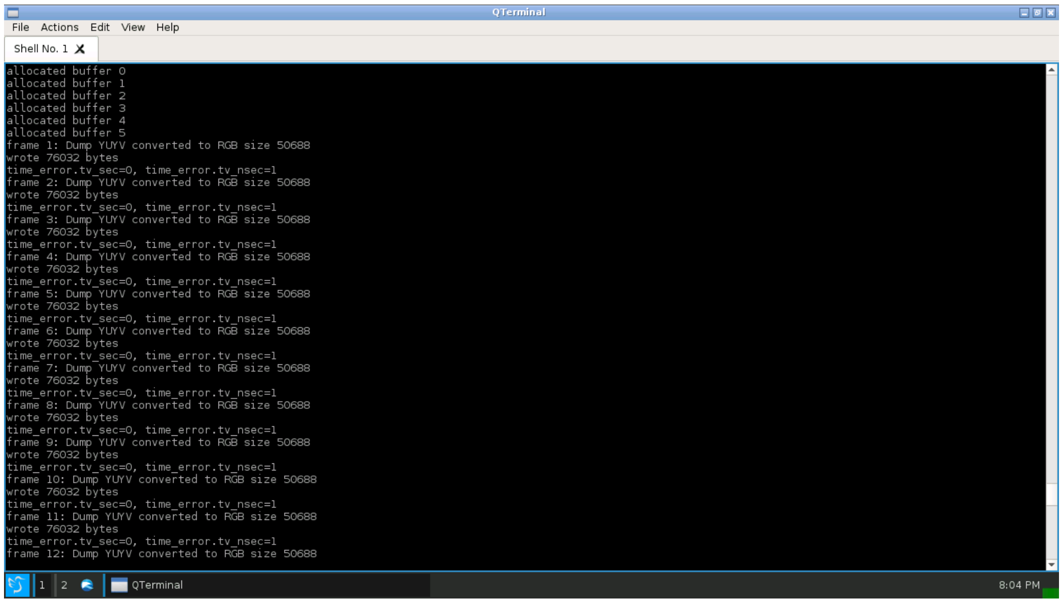
A picture was taken using the application and saved on the device.

Question 3

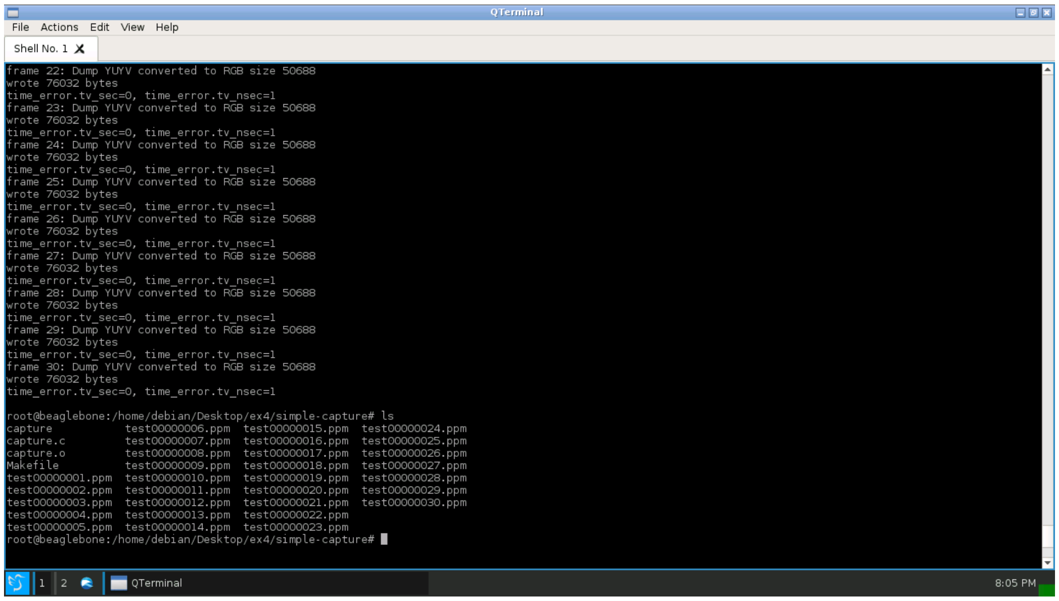
Using your verified Logitech C200 camera on a BeagleBone and we verify that it can stream continuously using to a raw image buffer for transformation and processing using example codes.

Simple-Capture:

This program was provided with V4L2 API and adapted by Prof. Siewert. This program uses the UVC drivers to acquire digital video from a source and timestamp each acquired frame saving it as a ppm file. We run the code, and observe the following:



```
allocated buffer 0
allocated buffer 1
allocated buffer 2
allocated buffer 3
allocated buffer 4
allocated buffer 5
frame 1: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 2: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 3: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 4: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 5: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 6: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 7: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 8: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 9: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 10: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 11: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 12: Dump YUYV converted to RGB size 50688
```



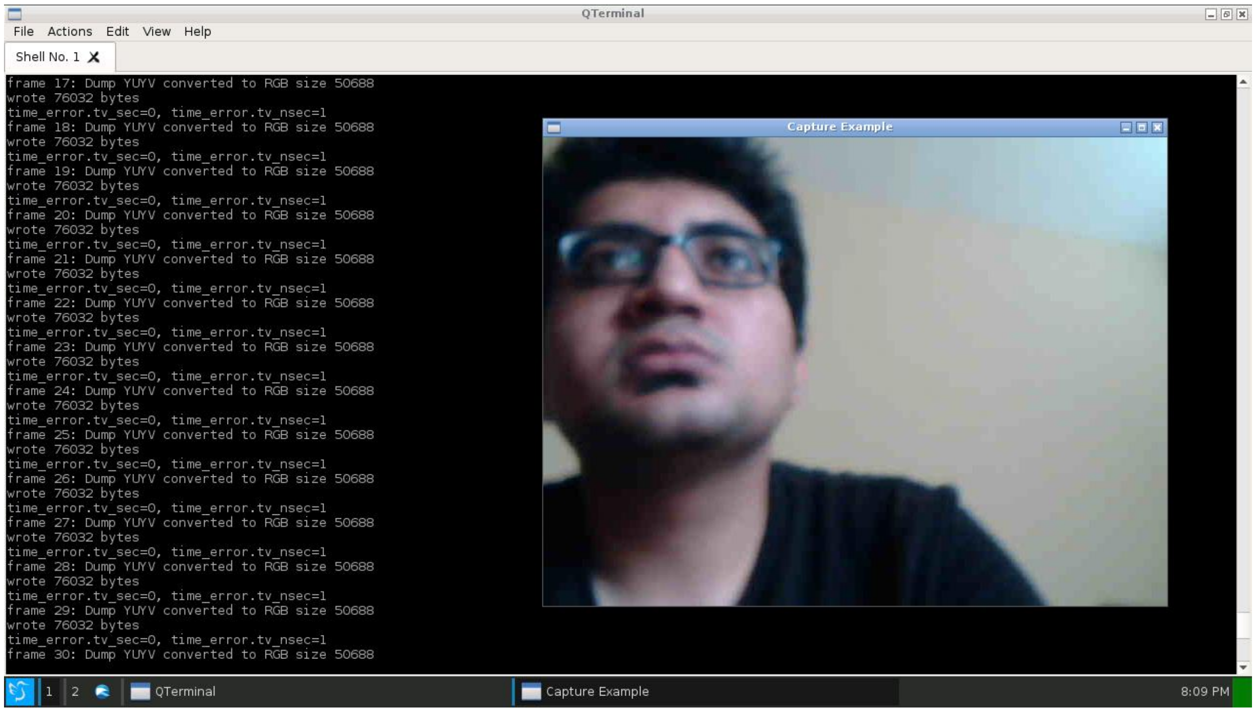
```
frame 22: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 23: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 24: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 25: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 26: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 27: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 28: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 29: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 30: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1

root@beaglebone:/home/debian/Desktop/ex4/simple-capture# ls
capture           test00000006.ppm  test00000015.ppm  test00000024.ppm
capture.c         test00000007.ppm  test00000016.ppm  test00000025.ppm
capture.o         test00000008.ppm  test00000017.ppm  test00000026.ppm
Makefile          test00000009.ppm  test00000018.ppm  test00000027.ppm
test00000001.ppm  test00000010.ppm  test00000019.ppm  test00000028.ppm
test00000002.ppm  test00000011.ppm  test00000020.ppm  test00000029.ppm
test00000003.ppm  test00000012.ppm  test00000021.ppm  test00000030.ppm
test00000004.ppm  test00000013.ppm  test00000022.ppm
test00000005.ppm  test00000014.ppm  test00000023.ppm
root@beaglebone:/home/debian/Desktop/ex4/simple-capture# █
```

We observe how the code captures 30 frames and timestamps and saves them as ppm files in the same directory.

Simpler-capture:

This program makes use of the opencv API to achieve the same goal of being able to stream and capture continuous media from the camera.

A window named “Capture example” is created using the cvNamedWindow() function. Next we use the cvCreateCameraCapture() function to start capturing video data as frames. Successive frames are requested using the cvQuerryFrame() function. This function returns a IplImage object. This frame image is then displayed onto the window using cvShowImage() function. Finally, the code checks for ESC keypress to terminate. This continues and hence giving us continuous video streaming effect. The screenshot of its execution is shown below:

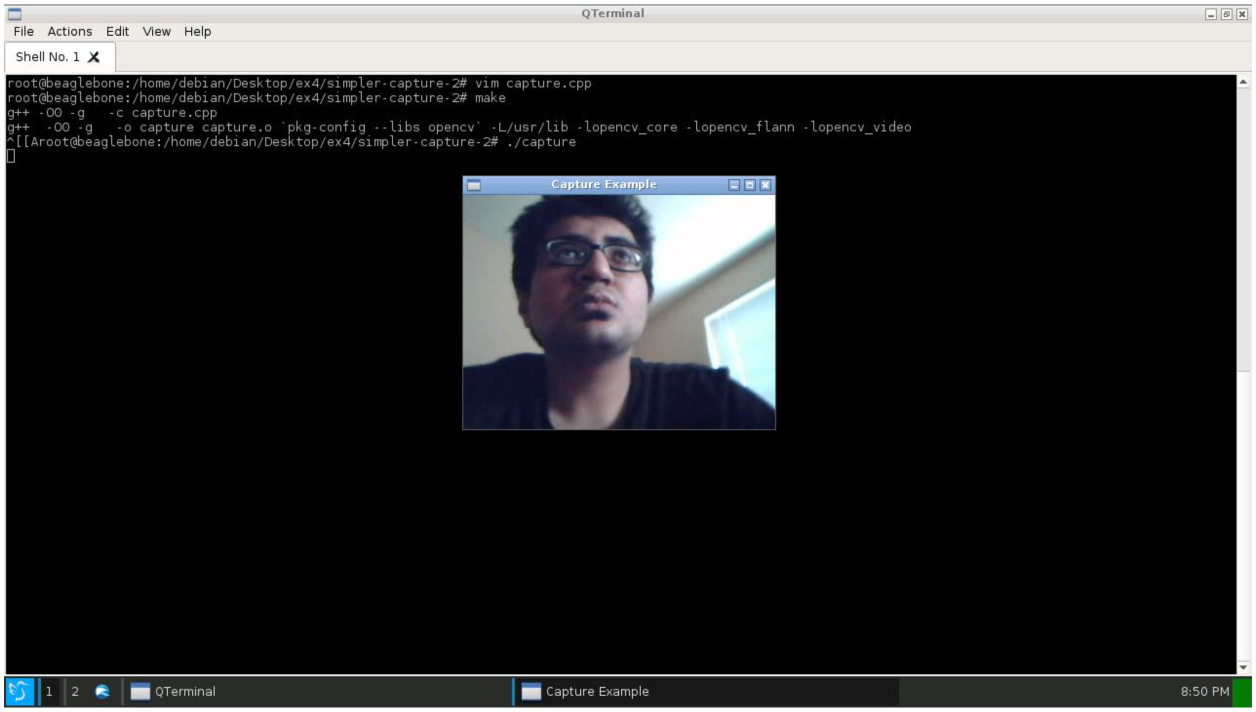

QTerminal
File Actions Edit View Help
Shell No. 1
Capture Example
1 2 QTerminal Capture Example 8:09 PM

```
frame 17: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 18: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 19: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 20: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 21: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 22: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 23: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 24: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 25: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 26: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 27: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 28: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 29: Dump YUYV converted to RGB size 50688
wrote 76032 bytes
time_error.tv_sec=0, time_error.tv_nsec=1
frame 30: Dump YUYV converted to RGB size 50688
```

Simpler-capture-2:

This code is similar to the previous simple-capture code. This program too utilizes the opencv API for continuous image capture. Only difference in this code is that we can set the resolution of the image captured. This is done by using the function cvSetCaptureProperty() function where we pass the desired height and width of captured frame as arguments.

I changed (lowered) the resolution and obtained an output as shown below:



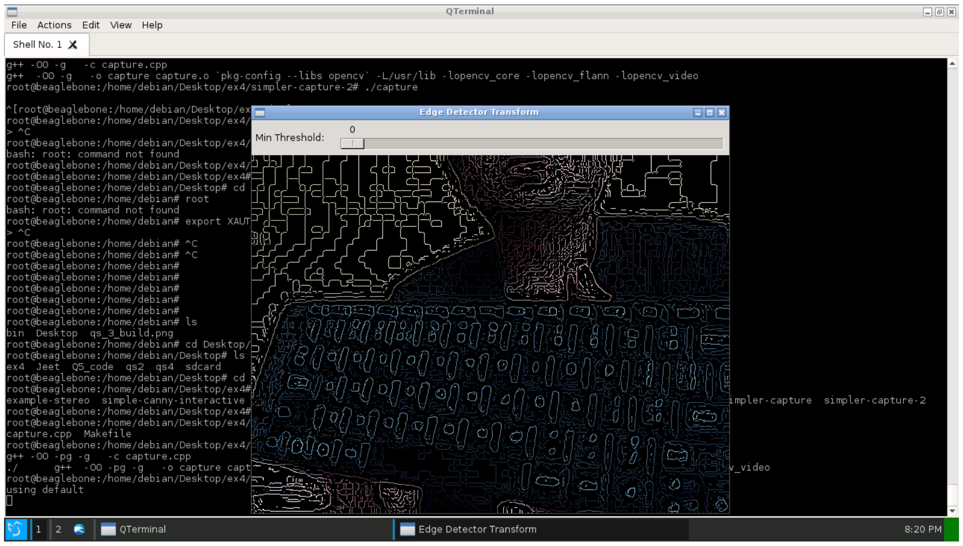
```
root@beaglebone:/home/debian/Desktop/ex4/simpler-capture-2# vim capture.cpp
root@beaglebone:/home/debian/Desktop/ex4/simpler-capture-2# make
g++ -O0 -g   -c capture.cpp
g++  -O0 -g   -o capture capture.o `pkg-config --libs opencv` -L/usr/lib -lopencv_core -lopencv_flann -lopencv_video
^[[Aroot@beaglebone:/home/debian/Desktop/ex4/simpler-capture-2# ./capture
```

Question 4

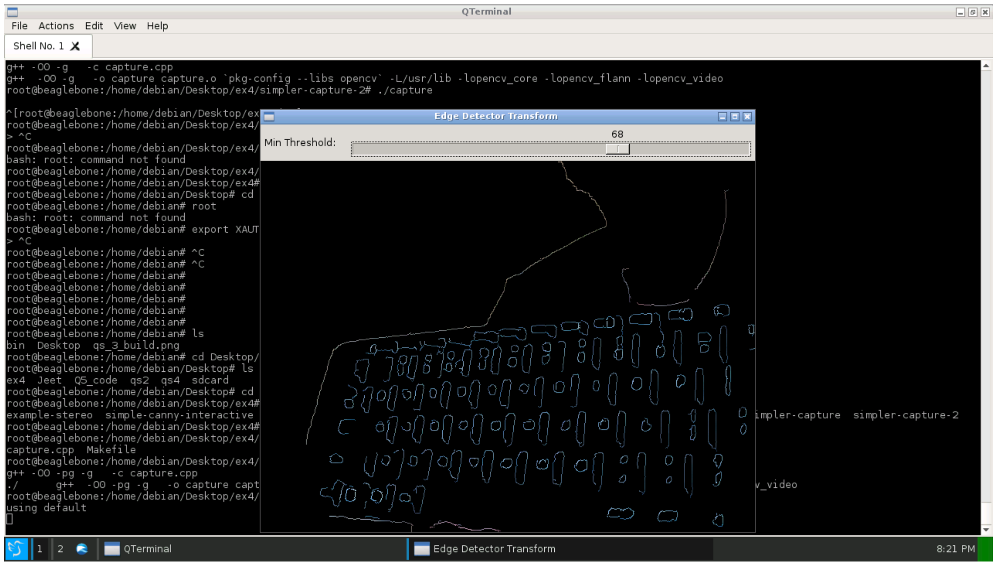
We now run continuous transformation opencv example codes.

Canny-interactive:

The canny-interactive is an edge detection algorithm which aims to have a low error rate, good localization, i.e., the distance between edge pixels detected and real edge pixels have to be minimized, and minimal response, i.e., only one detector response per edge. The following two images show the canny algorithm applied to similar images but with different lower threshold value. The lower threshold value is the highest pixel gradient which is rejected.



```
g++ -O0 -g   -c capture.cpp
g++  -O0 -g   -o capture capture.o `pkg-config --libs opencv` -L/usr/lib -lopencv_core -lopencv_flann -lopencv_video
root@beaglebone:/home/debian/Desktop/ex4/simpler-capture-2# ./capture

^[root@beaglebone:/home/debian/Desktop/ex
root@beaglebone:/home/debian/Desktop/ex4/
> ^C
root@beaglebone:/home/debian/Desktop/ex4/
bash: root: command not found
root@beaglebone:/home/debian/Desktop/ex4/
root@beaglebone:/home/debian/Desktop/ex4#
root@beaglebone:/home/debian/Desktop# cd
root@beaglebone:/home/debian# root
bash: root: command not found
root@beaglebone:/home/debian# export XAUT
> ^C
root@beaglebone:/home/debian# ^C
root@beaglebone:/home/debian# ^C
root@beaglebone:/home/debian#
root@beaglebone:/home/debian#
root@beaglebone:/home/debian#
root@beaglebone:/home/debian#
root@beaglebone:/home/debian#
root@beaglebone:/home/debian# ls
bin  Desktop  qs_3_build.png
root@beaglebone:/home/debian# cd Desktop/
root@beaglebone:/home/debian/Desktop# ls
ex4  Jeet  Q5_code  qs2  qs4  sdcard
root@beaglebone:/home/debian/Desktop# cd
root@beaglebone:/home/debian/Desktop/ex4#
example-stereo  simple-canny-interactive                                              impler-capture  simpler-capture-2
root@beaglebone:/home/debian/Desktop/ex4#
root@beaglebone:/home/debian/Desktop/ex4/
capture.cpp  Makefile
root@beaglebone:/home/debian/Desktop/ex4/
g++ -O0 -pg -g   -c capture.cpp
./      g++  -O0 -pg -g   -o capture capt                                                 v_video
root@beaglebone:/home/debian/Desktop/ex4/
using default
```

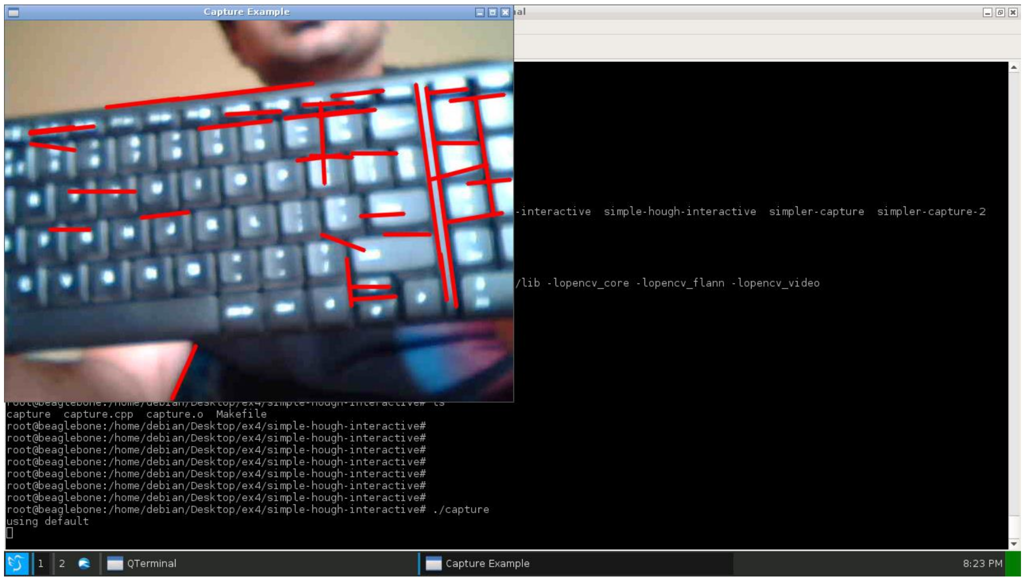


```
g++ -O0 -g   -c capture.cpp
g++  -O0 -g   -o capture capture.o `pkg-config --libs opencv` -L/usr/lib -lopencv_core -lopencv_flann -lopencv_video
root@beaglebone:/home/debian/Desktop/ex4/simpler-capture-2# ./capture

^[root@beaglebone:/home/debian/Desktop/ex
root@beaglebone:/home/debian/Desktop/ex4/
> ^C
root@beaglebone:/home/debian/Desktop/ex4/
bash: root: command not found
root@beaglebone:/home/debian/Desktop/ex4/
root@beaglebone:/home/debian/Desktop/ex4#
root@beaglebone:/home/debian/Desktop# cd
root@beaglebone:/home/debian# root
bash: root: command not found
root@beaglebone:/home/debian# export XAUT
> ^C
root@beaglebone:/home/debian# ^C
root@beaglebone:/home/debian# ^C
root@beaglebone:/home/debian#
root@beaglebone:/home/debian#
root@beaglebone:/home/debian#
root@beaglebone:/home/debian#
root@beaglebone:/home/debian#
root@beaglebone:/home/debian# ls
bin  Desktop  qs_3_build.png
root@beaglebone:/home/debian# cd Desktop/
root@beaglebone:/home/debian/Desktop# ls
ex4  Jeet  Q5_code  qs2  qs4  sdcard
root@beaglebone:/home/debian/Desktop# cd
root@beaglebone:/home/debian/Desktop/ex4#
example-stereo  simple-canny-interactive                                              impler-capture  simpler-capture-2
root@beaglebone:/home/debian/Desktop/ex4#
root@beaglebone:/home/debian/Desktop/ex4/
capture.cpp  Makefile
root@beaglebone:/home/debian/Desktop/ex4/
g++ -O0 -pg -g   -c capture.cpp
./      g++  -O0 -pg -g   -o capture capt                                             v_video
root@beaglebone:/home/debian/Desktop/ex4/
using default
```

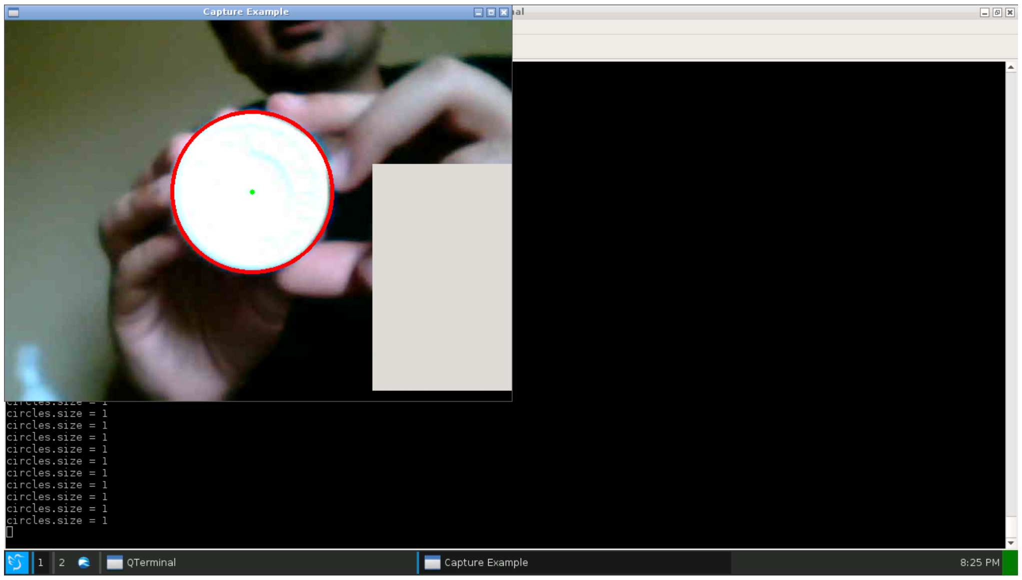
Code Explanation:

First the code generates a window with a trackbar for the user to enter the minimum threshold. Then we start streaming continuous data using the function cvCreateCameraCapture(). Next we continuously obtain frame using the cvQuerryFrame() function and call the canny threshold function for each frame. This function converts the image to grey scale and reduces noise with a kernel size 3. The opencv function canny is then applied. Here the destination image is first filled with 0s (black), the image from the output of the canny function (edges detected) is copied onto the destination image hence giving us an image with only the edges. This image is displayed on the window. Finally, we poll for a keypress, which if is ESC we terminate the window.

I also experimented with Hough-line and Hough-elliptical transforms. The Hough-line transform continuously identifies straight lines in the image and the Hough-elliptical transform locates circles in the image. The output screenshots for both are as shown below:



```
-interactive  simple-hough-interactive  simpler-capture  simpler-capture-2

/lib -lopencv_core -lopencv_flann -lopencv_video

capture  capture.cpp  capture.o  Makefile
root@beaglebone:/home/debian/Desktop/ex4/simple-hough-interactive#
root@beaglebone:/home/debian/Desktop/ex4/simple-hough-interactive#
root@beaglebone:/home/debian/Desktop/ex4/simple-hough-interactive#
root@beaglebone:/home/debian/Desktop/ex4/simple-hough-interactive#
root@beaglebone:/home/debian/Desktop/ex4/simple-hough-interactive#
root@beaglebone:/home/debian/Desktop/ex4/simple-hough-interactive#
root@beaglebone:/home/debian/Desktop/ex4/simple-hough-interactive#
root@beaglebone:/home/debian/Desktop/ex4/simple-hough-interactive# ./capture
using default
```



```
circles.size = 1
circles.size = 1
circles.size = 1
circles.size = 1
circles.size = 1
circles.size = 1
circles.size = 1
circles.size = 1
circles.size = 1
circles.size = 1
```

Applications of Canny Edge Detection:

Canny's algorithm for edge detection has very diverse applications from biology to civil engineering. A few applications I came across are listed below:

- Used for monitoring change in shape and size of cells. Canny's algorithm by identifying edges, identifies shape and size of object in the image. This idea has been used to monitor change in shape and size of biological cells when subjected to varying conditions.
- Continuous transform edge detection can also be used identify cracks in structures or roads.
- Continuous transform applications can also be used on video to track movement of objects which has applications in surveillance, solar tracking, and ball tracking in sports.
- Canny Transform can also be used for tracking specific edges like straight lines and circles as applied in Hough transforms.

Question 5

In this question I have collaborated with Akshay Iyer and Sayan Barman for capturing data.

A code was developed to compare the average frame rate by applying three different transformations for different resolutions. To estimate the deadline, we first determined the time taken for a frame to be processed in each of the transformations individually by calculating the average frame rate.

Algorithm Analysis:

I have used transformations of canny threshold, Hough Line Interactive and Hough Elliptical. In each of these transformations, 100 frames were processed and the total time taken to process these 100 frames was determined in milliseconds. Using this time, average time taken to process single frame was determined in each of the three transformations. Soft real-time deadline was chosen by taking the summation of the time taken for all three transformations. I have chosen against adding a margin to the obtained sum because on execution of my prototype described further I noticed that some resolutions resulted in positive jitter and some resulted in negative jitter. Addition of a margin time shall only worsen the situation of some and better for other. Hence I have chosen to not add a margin and be as close to true total execution times for each resolution. The average time taken to process each frame in each of the three transformations and the deadline chosen is as shown below.

	160 x 120	**320 x 240**	**480 x 360**	**640 x 480**
Canny Threshold	112.05	126.79	131.58	208.94
Hough Line	109.15	128.11	139.50	247.60
Hough Elliptical	109.41	124.16	128.75	202.66
Total Deadline	330	380	400	660
Execution Time	377.66	393.76	395.01	684.11
Jitter	47.66	13.76	-4.99	24.11

Prototype Analysis:

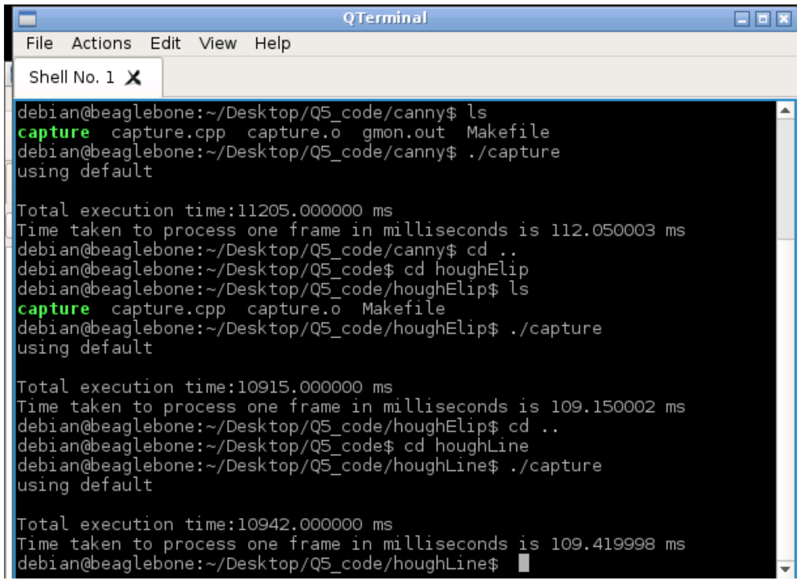
Now, we write a code in which three threads are spawned where each of the threads perform a transformation using SCHED_FIFO scheduling policy. Each transformation pertaining to a thread is performed for 100 frames. To perform this transformation sequentially, three binary semaphores are used in this code. The main function initially makes the semaphore for canny edge detector thread available, due to which the canny edge detector thread runs. When the canny edge detector thread finishes execution, it releases the semaphore for Hough line thread, which runs to completion and releases semaphore for Hough elliptical thread. The Hough eliptical thread runs to completion and releases the semaphore for canny edge detector thread. The loop continues for 100 iterations. As the camera is a shared resource, a mutex lock is done before capturing the frame in each of the transformations and unlocked after capturing the frame. The execution time which is the average time taken to process a frame in each of the three transformation is determined and is shown in the table above.

Jitter Analysis:

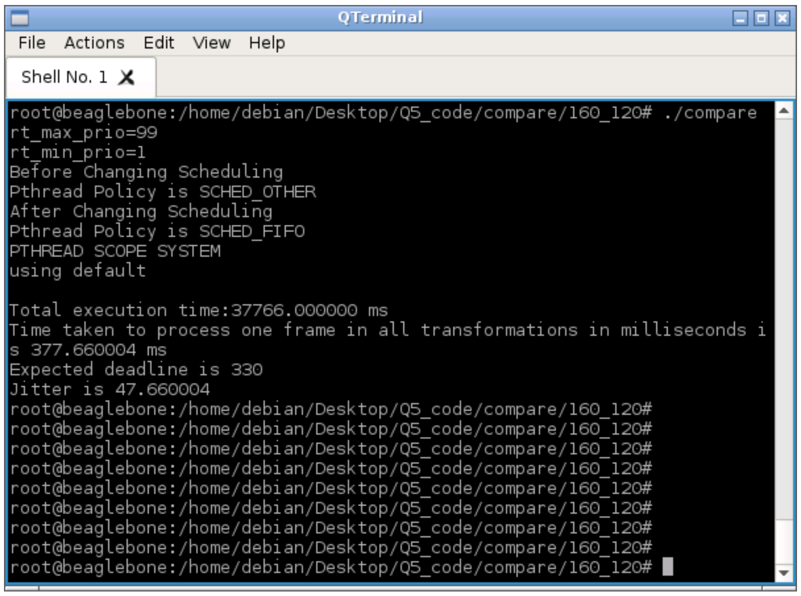
Jitter relative to the deadline is calculated by taking the difference between the average execution time and the deadline (determined as mentioned above). This process is repeated for all the resolutions. Jitter is negative if the execution is completed before the deadline. The jitter is positive if the time taken to execute is beyond the deadline. The resolutions that we have considered is 640*480, 480*360, 320*240, 160*120. The average time taken to process a frame to calculate the soft real-time deadline and the average time taken to process a frame to determine jitter may vary as time taken depends on the view of the camera. For Example, average time taken to perform Hough Elliptical transformations may increase if there are more number elliptical shaped objects in the view. We also see that the processing time decreases as we decrease the resolution of the images.

Screenshots:

For 160 x 120:

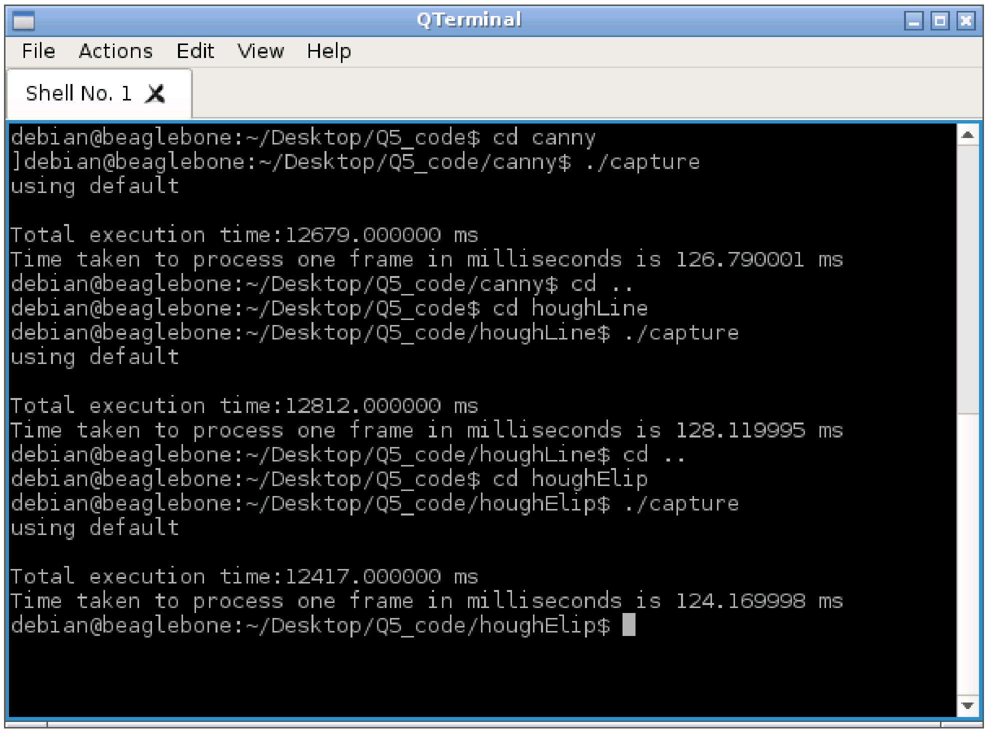


```
debian@beaglebone:~/Desktop/Q5_code/canny$ ls
capture  capture.cpp  capture.o  gmon.out  Makefile
debian@beaglebone:~/Desktop/Q5_code/canny$ ./capture
using default

Total execution time:11205.000000 ms
Time taken to process one frame in milliseconds is 112.050003 ms
debian@beaglebone:~/Desktop/Q5_code/canny$ cd ..
debian@beaglebone:~/Desktop/Q5_code$ cd houghElip
debian@beaglebone:~/Desktop/Q5_code/houghElip$ ls
capture  capture.cpp  capture.o  Makefile
debian@beaglebone:~/Desktop/Q5_code/houghElip$ ./capture
using default

Total execution time:10915.000000 ms
Time taken to process one frame in milliseconds is 109.150002 ms
debian@beaglebone:~/Desktop/Q5_code/houghElip$ cd ..
debian@beaglebone:~/Desktop/Q5_code$ cd houghLine
debian@beaglebone:~/Desktop/Q5_code/houghLine$ ./capture
using default

Total execution time:10942.000000 ms
Time taken to process one frame in milliseconds is 109.419998 ms
debian@beaglebone:~/Desktop/Q5_code/houghLine$
```

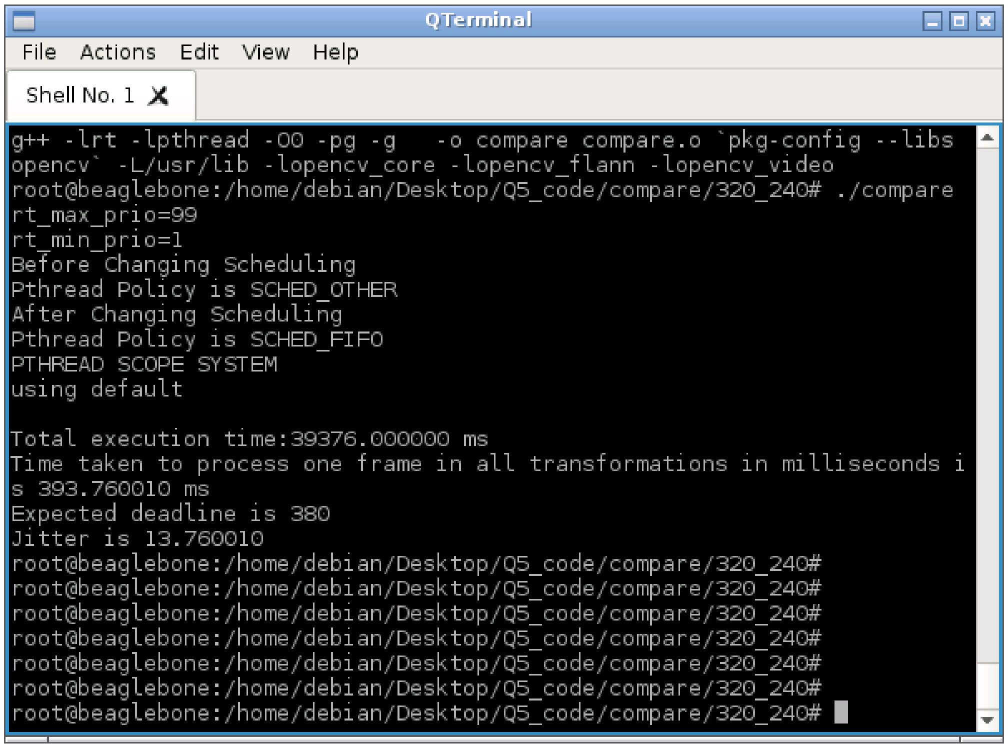


```
root@beaglebone:/home/debian/Desktop/Q5_code/compare/160_120# ./compare
rt_max_prio=99
rt_min_prio=1
Before Changing Scheduling
Pthread Policy is SCHED_OTHER
After Changing Scheduling
Pthread Policy is SCHED_FIFO
PTHREAD SCOPE SYSTEM
using default

Total execution time:37766.000000 ms
Time taken to process one frame in all transformations in milliseconds i
s 377.660004 ms
Expected deadline is 330
Jitter is 47.660004
root@beaglebone:/home/debian/Desktop/Q5_code/compare/160_120#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/160_120#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/160_120#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/160_120#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/160_120#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/160_120#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/160_120#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/160_120#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/160_120#
```

For 320 x 240:

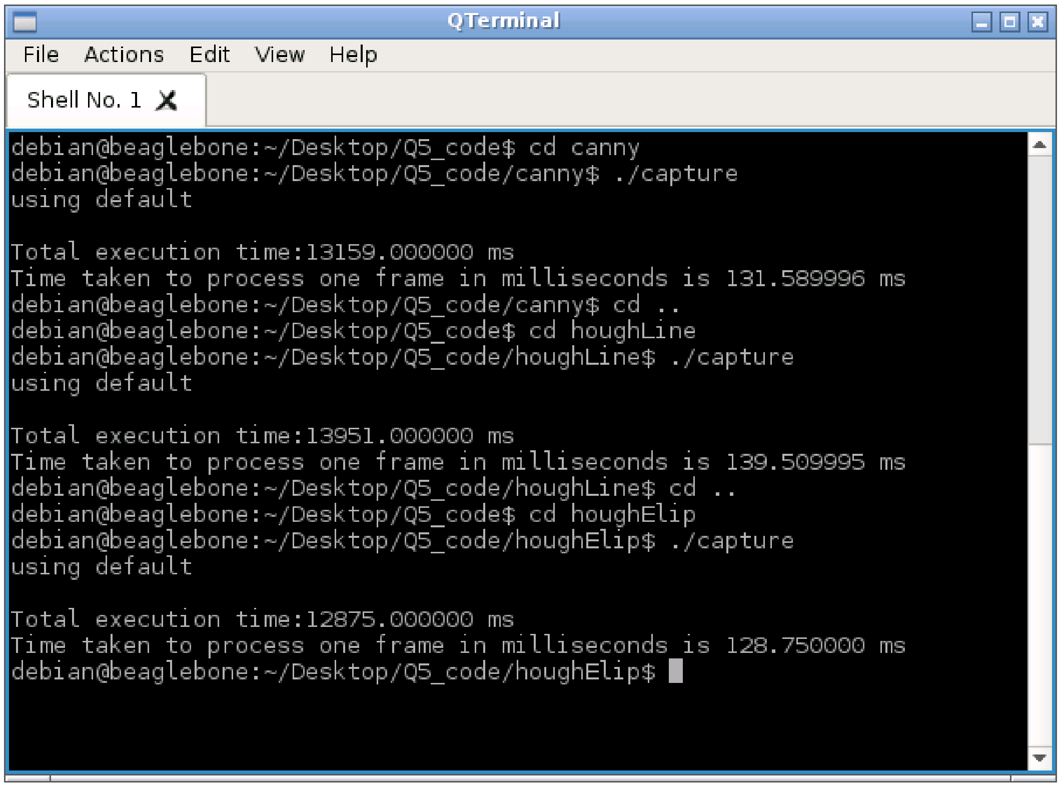


```
debian@beaglebone:~/Desktop/Q5_code$ cd canny
]debian@beaglebone:~/Desktop/Q5_code/canny$ ./capture
using default

Total execution time:12679.000000 ms
Time taken to process one frame in milliseconds is 126.790001 ms
debian@beaglebone:~/Desktop/Q5_code/canny$ cd ..
debian@beaglebone:~/Desktop/Q5_code$ cd houghLine
debian@beaglebone:~/Desktop/Q5_code/houghLine$ ./capture
using default

Total execution time:12812.000000 ms
Time taken to process one frame in milliseconds is 128.119995 ms
debian@beaglebone:~/Desktop/Q5_code/houghLine$ cd ..
debian@beaglebone:~/Desktop/Q5_code$ cd houghElip
debian@beaglebone:~/Desktop/Q5_code/houghElip$ ./capture
using default

Total execution time:12417.000000 ms
Time taken to process one frame in milliseconds is 124.169998 ms
debian@beaglebone:~/Desktop/Q5_code/houghElip$
```

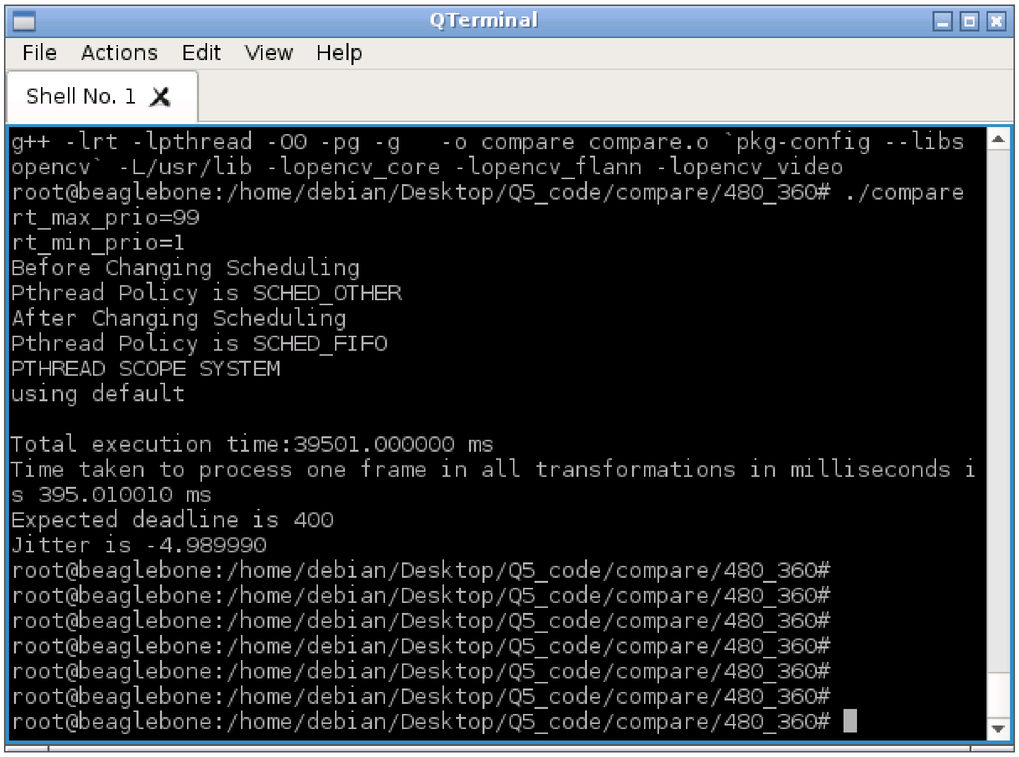


```
g++ -lrt -lpthread -O0 -pg -g   -o compare compare.o `pkg-config --libs
opencv` -L/usr/lib -lopencv_core -lopencv_flann -lopencv_video
root@beaglebone:/home/debian/Desktop/Q5_code/compare/320_240# ./compare
rt_max_prio=99
rt_min_prio=1
Before Changing Scheduling
Pthread Policy is SCHED_OTHER
After Changing Scheduling
Pthread Policy is SCHED_FIFO
PTHREAD SCOPE SYSTEM
using default

Total execution time:39376.000000 ms
Time taken to process one frame in all transformations in milliseconds i
s 393.760010 ms
Expected deadline is 380
Jitter is 13.760010
root@beaglebone:/home/debian/Desktop/Q5_code/compare/320_240#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/320_240#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/320_240#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/320_240#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/320_240#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/320_240#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/320_240#
```

For 480 x 360

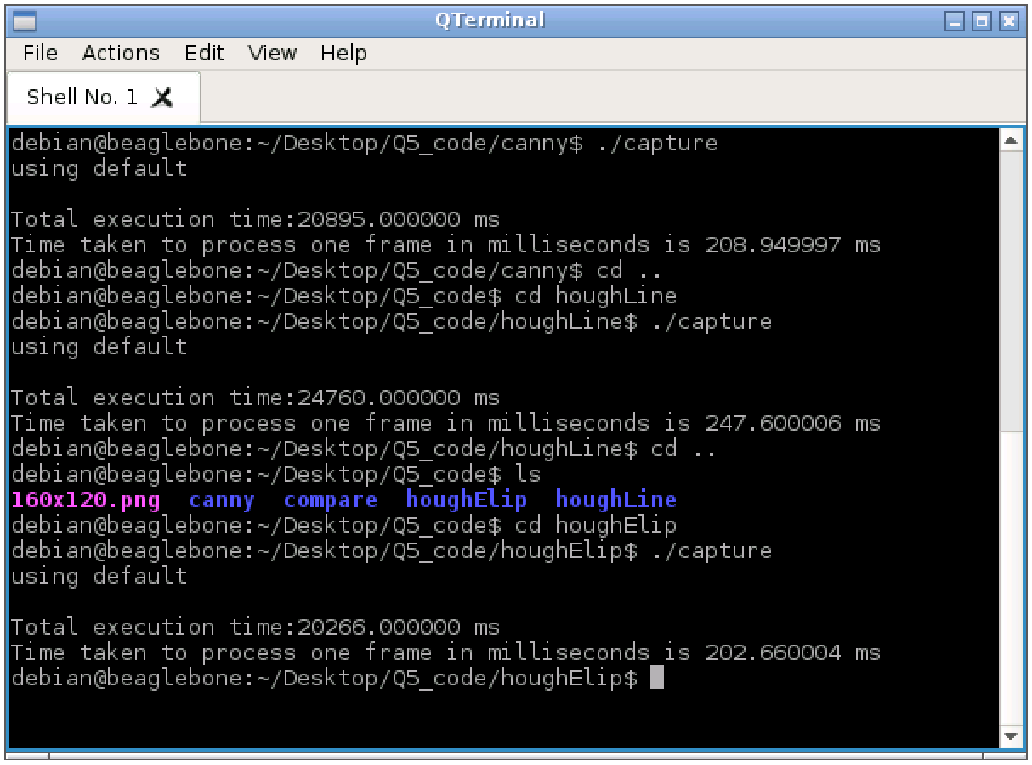


```
debian@beaglebone:~/Desktop/Q5_code$ cd canny
debian@beaglebone:~/Desktop/Q5_code/canny$ ./capture
using default

Total execution time:13159.000000 ms
Time taken to process one frame in milliseconds is 131.589996 ms
debian@beaglebone:~/Desktop/Q5_code/canny$ cd ..
debian@beaglebone:~/Desktop/Q5_code$ cd houghLine
debian@beaglebone:~/Desktop/Q5_code/houghLine$ ./capture
using default

Total execution time:13951.000000 ms
Time taken to process one frame in milliseconds is 139.509995 ms
debian@beaglebone:~/Desktop/Q5_code/houghLine$ cd ..
debian@beaglebone:~/Desktop/Q5_code$ cd houghElip
debian@beaglebone:~/Desktop/Q5_code/houghElip$ ./capture
using default

Total execution time:12875.000000 ms
Time taken to process one frame in milliseconds is 128.750000 ms
debian@beaglebone:~/Desktop/Q5_code/houghElip$
```

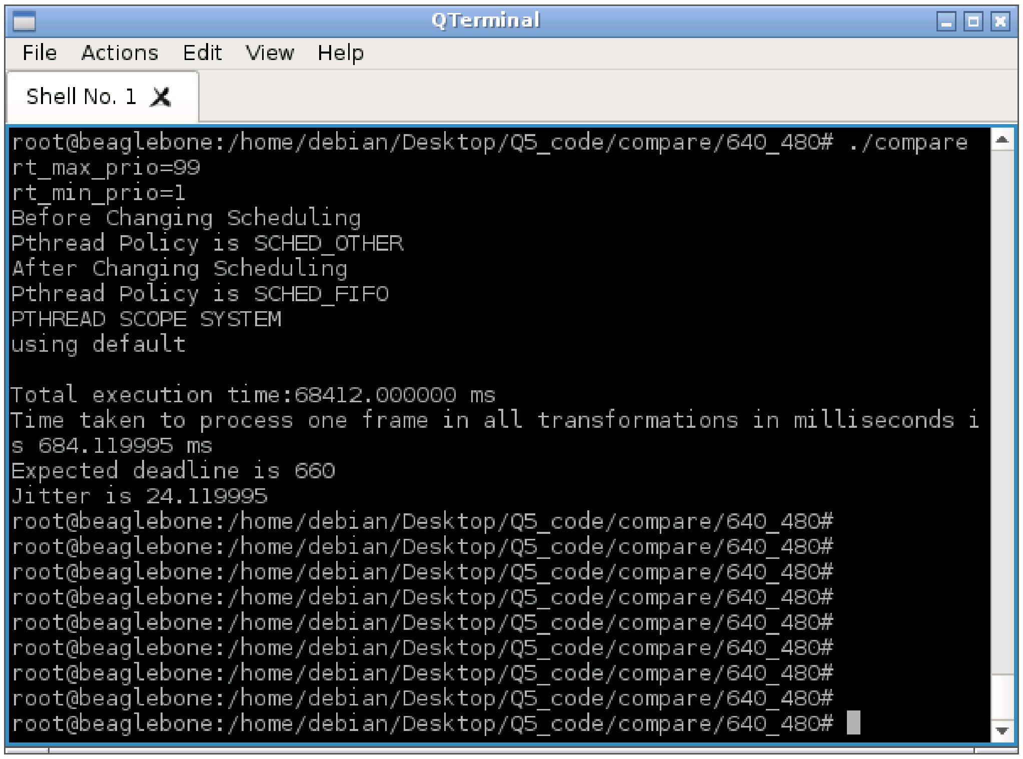


```
g++ -lrt -lpthread -O0 -pg -g   -o compare compare.o `pkg-config --libs
opencv` -L/usr/lib -lopencv_core -lopencv_flann -lopencv_video
root@beaglebone:/home/debian/Desktop/Q5_code/compare/480_360# ./compare
rt_max_prio=99
rt_min_prio=1
Before Changing Scheduling
Pthread Policy is SCHED_OTHER
After Changing Scheduling
Pthread Policy is SCHED_FIFO
PTHREAD SCOPE SYSTEM
using default

Total execution time:39501.000000 ms
Time taken to process one frame in all transformations in milliseconds i
s 395.010010 ms
Expected deadline is 400
Jitter is -4.989990
root@beaglebone:/home/debian/Desktop/Q5_code/compare/480_360#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/480_360#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/480_360#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/480_360#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/480_360#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/480_360#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/480_360#
```

For 640 x 480

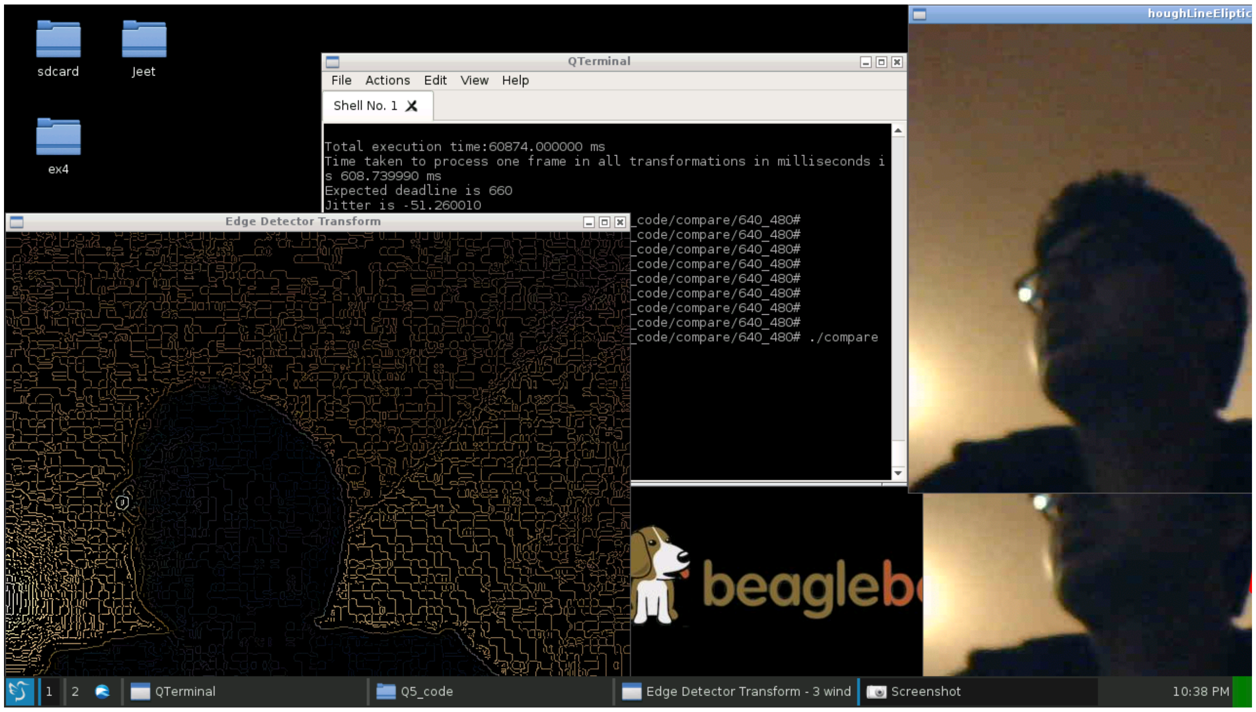


```
debian@beaglebone:~/Desktop/Q5_code/canny$ ./capture
using default

Total execution time:20895.000000 ms
Time taken to process one frame in milliseconds is 208.949997 ms
debian@beaglebone:~/Desktop/Q5_code/canny$ cd ..
debian@beaglebone:~/Desktop/Q5_code$ cd houghLine
debian@beaglebone:~/Desktop/Q5_code/houghLine$ ./capture
using default

Total execution time:24760.000000 ms
Time taken to process one frame in milliseconds is 247.600006 ms
debian@beaglebone:~/Desktop/Q5_code/houghLine$ cd ..
debian@beaglebone:~/Desktop/Q5_code$ ls
160x120.png  canny  compare  houghElip  houghLine
debian@beaglebone:~/Desktop/Q5_code$ cd houghElip
debian@beaglebone:~/Desktop/Q5_code/houghElip$ ./capture
using default

Total execution time:20266.000000 ms
Time taken to process one frame in milliseconds is 202.660004 ms
debian@beaglebone:~/Desktop/Q5_code/houghElip$
```



```
root@beaglebone:/home/debian/Desktop/Q5_code/compare/640_480# ./compare
rt_max_prio=99
rt_min_prio=1
Before Changing Scheduling
Pthread Policy is SCHED_OTHER
After Changing Scheduling
Pthread Policy is SCHED_FIFO
PTHREAD SCOPE SYSTEM
using default

Total execution time:68412.000000 ms
Time taken to process one frame in all transformations in milliseconds i
s 684.119995 ms
Expected deadline is 660
Jitter is 24.119995
root@beaglebone:/home/debian/Desktop/Q5_code/compare/640_480#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/640_480#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/640_480#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/640_480#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/640_480#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/640_480#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/640_480#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/640_480#
root@beaglebone:/home/debian/Desktop/Q5_code/compare/640_480#
```

Executing all three transforms simultaneously as threads:



```
Total execution time:60874.000000 ms
Time taken to process one frame in all transformations in milliseconds i
s 608.739990 ms
Expected deadline is 660
Jitter is -51.260010
_code/compare/640_480#
_code/compare/640_480#
_code/compare/640_480#
_code/compare/640_480#
_code/compare/640_480#
_code/compare/640_480#
_code/compare/640_480#
_code/compare/640_480#
_code/compare/640_480# ./compare
```