# ECEN5623, Real-Time Embedded Systems

# Exercise #4- Real-Time Continuous Media

UNDER THE GUIDANCE OF:

PROFESSOR TIMOTHY SCHERR

University of Colorado Boulder

SUBMITTED BY:

VAISHNAVI KULKARNI

ATHARV DESAI

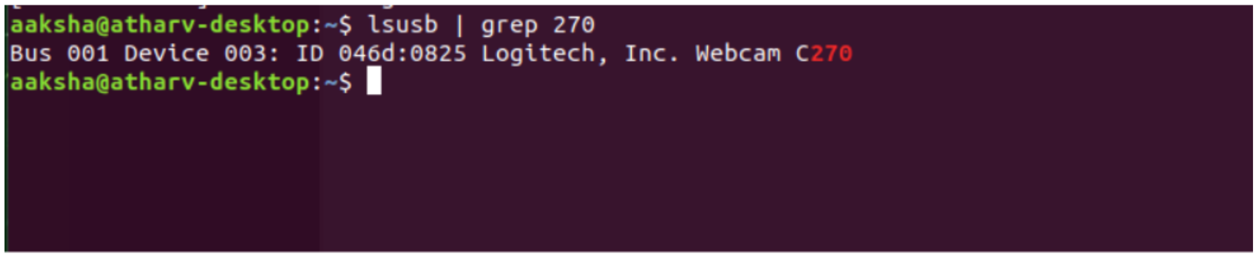
DEVELOPMENT PLATFORM: JETSON NANO

# Exercise 4: Real-Time Continuous Media

1) [10 points] Obtain a Logitech C200 camera or equivalent and verify that is detected by the DE1-SoC, Raspberry Pi or Jetson Board USB driver. You can check the camera out from the TA's or purchase one of your own, or use another camera that has a compliant UVC driver. Use lsusb, lsmod and dmesg kernel driver configuration tool to make sure your Logitech C200 USB camera is plugged in and recognized by your DE1-SoC, Raspberry Pi or Jetson (note that on the Jetson, it does not use driver modules, but rather a monolithic kernel image, so lsmod will not look the same as other Linux systems – see what you can find by exploring /cat/proc on you Jetson to find the camera USB device). For the Jetson, do lsusb | grep C200 and prove to the TA (and more importantly yourself) with that output (screenshot) that your camera is recognized. For systems other than a Jetson, do lsmod | grep video and verify that the UVC driver is loaded as well (http://www.ideasonboard.org/uvc/ ). To further verify, or debug if you don't see the UVC driver loaded in response to plugging in the USB camera, do dmesg | grep video or just dmesg and scroll through the log messages to see if your USB device was found. Capture all output and annotate what you see with descriptions to the best of your understanding

<u>Solution</u>:

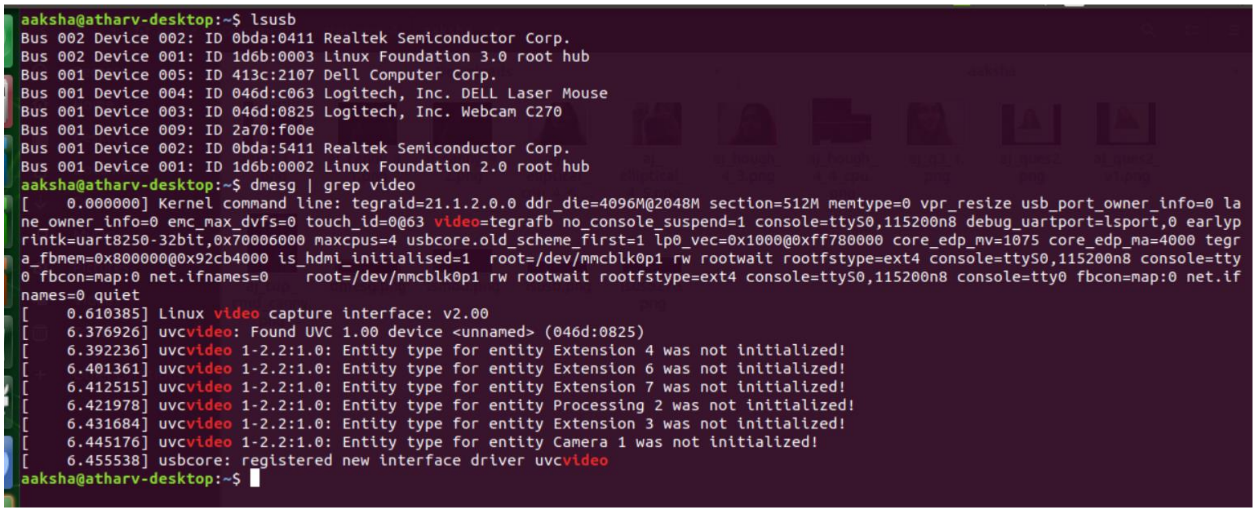
➔ For this project, We interfaced Jetson Nano with Logitech Camera C270. The below screenshot shows that the camera was recognized by the Jetson Nano



```
aaksha@atharv-desktop:~$ lsusb | grep 270
Bus 001 Device 003: ID 046d:0825 Logitech, Inc. Webcam C270
aaksha@atharv-desktop:~$ 
```

Figure 1: lsusb command to show camera interfacing

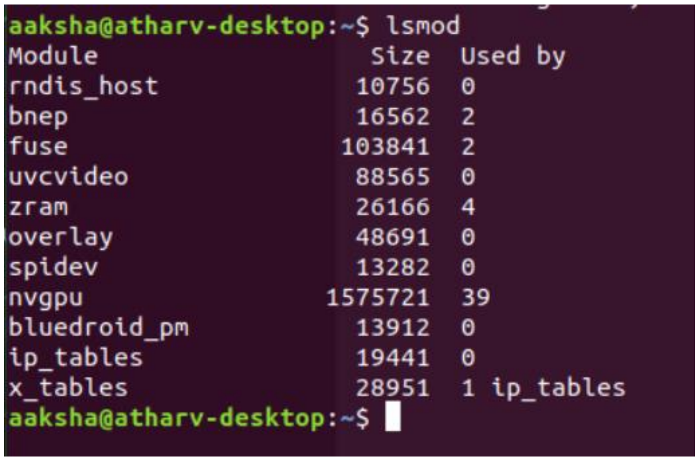
➔ The camera is interfaced at Bus 001 and Device Number assigned is 003.
➔ The ' lsusb | grep 270 ' command can be used to search the camera and show it on the terminal.



```
aaksha@atharv-desktop:~$ lsusb
Bus 002 Device 002: ID 0bda:0411 Realtek Semiconductor Corp.
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 005: ID 413c:2107 Dell Computer Corp.
Bus 001 Device 004: ID 046d:c063 Logitech, Inc. DELL Laser Mouse
Bus 001 Device 003: ID 046d:0825 Logitech, Inc. Webcam C270
Bus 001 Device 009: ID 2a70:f00e
Bus 001 Device 002: ID 0bda:5411 Realtek Semiconductor Corp.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
aaksha@atharv-desktop:~$ dmesg | grep video
[    0.000000] Kernel command line: tegraid=21.1.2.0.0 ddr_die=4096M@2048M section=512M memtype=0 vpr_resize usb_port_owner_info=0 la
ne_owner_info=0 emc_max_dvfs=0 touch_id=0@63 video=tegrafb no_console_suspend=1 console=ttyS0,115200n8 debug_uartport=lsport,0 earlyp
rintk=uart8250-32bit,0x70006000 maxcpus=4 usbcore.old_scheme_first=1 lp0_vec=0x1000@0xff780000 core_edp_mv=1075 core_edp_ma=4000 tegr
a_fbmem=0x800000@0x92cb4000 is_hdmi_initialised=1  root=/dev/mmcblk0p1 rw rootwait rootfstype=ext4 console=ttyS0,115200n8 console=tty
0 fbcon=map:0 net.ifnames=0    root=/dev/mmcblk0p1 rw rootwait rootfstype=ext4 console=ttyS0,115200n8 console=tty0 fbcon=map:0 net.if
names=0 quiet
[    0.610385] Linux video capture interface: v2.00
[    6.376926] uvcvideo: Found UVC 1.00 device <unnamed> (046d:0825)
[    6.392236] uvcvideo 1-2.2:1.0: Entity type for entity Extension 4 was not initialized!
[    6.401361] uvcvideo 1-2.2:1.0: Entity type for entity Extension 6 was not initialized!
[    6.412515] uvcvideo 1-2.2:1.0: Entity type for entity Extension 7 was not initialized!
[    6.421978] uvcvideo 1-2.2:1.0: Entity type for entity Processing 2 was not initialized!
[    6.431684] uvcvideo 1-2.2:1.0: Entity type for entity Extension 3 was not initialized!
[    6.445176] uvcvideo 1-2.2:1.0: Entity type for entity Camera 1 was not initialized!
[    6.455538] usbcore: registered new interface driver uvcvideo
aaksha@atharv-desktop:~$
```

Figure 2: dmesg command to see UVC Driver Loading

➔ On dmesg command, v2 linux video capture was used and UVC device



```
aaksha@atharv-desktop:~$ lsmod
Module                  Size  Used by
rndis_host             10756  0
bnep                   16562  2
fuse                  103841  2
uvcvideo               88565  0
zram                   26166  4
overlay                48691  0
spidev                 13282  0
nvgpu                1575721  39
bluedroid_pm           13912  0
ip_tables              19441  0
x_tables               28951  1 ip_tables
aaksha@atharv-desktop:~$
```

Figure 3: lsmod command to see UVC module size

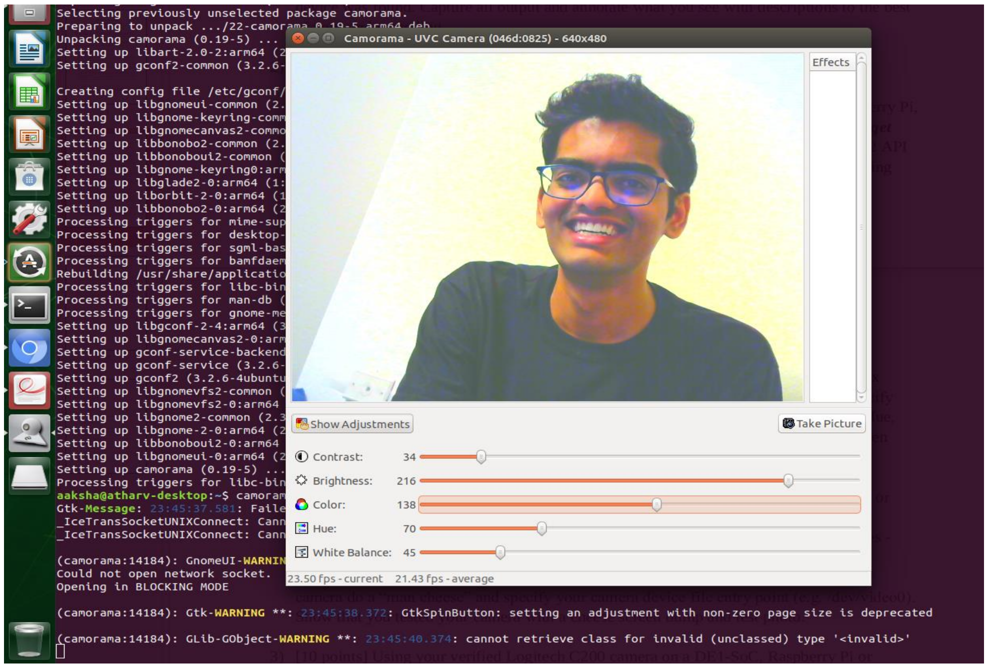
➔ The lsmod command was executed to show the size of UVCVideo module which was 88565.

2) [10 points]  Option 1:  Camorama If you do not have camorama, do apt-get install camorama on your DE1-SoC, Raspberry Pi, or Jetson board [you may need to first do sudo add-apt-repository universe; sudo apt-get update].  This should not only install nice camera capture GUI tools, but also the V4L2 API (described well in this series of Linux articles - http://lwn.net/Articles/203924/ ).  Running

camorama should provide an interactive camera control session for your Logitech C2xx camera – if you have issues connecting to your camera do a "man camorama" and specify your camera device file entry point (e.g. /dev/video0).  Run camorama and play with Hue, Color, Brightness, White Balance and Contrast, take an example image and take a screen shot of the tool and provide both in your report

<u>Solution</u>:

- ➔ For taking a single snapshot capture, we had to install either camorama or cheese tool.
- ➔ There are different linux camera softwares like Motion, Zoneminder, Cheese, Ekiga.
- ➔ We used camorama tool. Attached below are the screenshots of camorama tool which is GNOME2 Video4Linux Viewer and has features like contrast, hue, white balance.



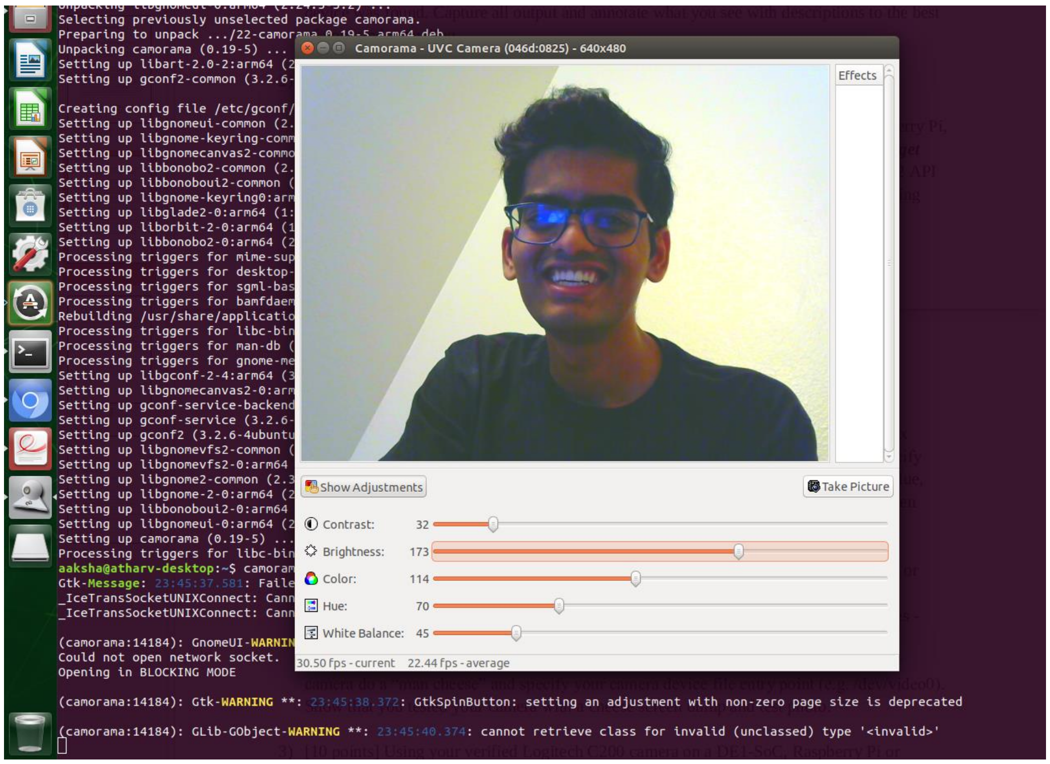
```
Selecting previously unselected package camorama.
Preparing to unpack .../22-camorama_0.19-5_arm64.deb
Unpacking camorama (0.19-5) ...
Setting up libart-2.0-2:arm64 (2
Setting up gconf2-common (3.2.6-

Creating config file /etc/gconf/
Setting up libgnomeui-common (2.
Setting up libgnome-keyring-comm
Setting up libgnomecanvas2-commo
Setting up libbonobo2-common (2.
Setting up libbonoboui2-common (
Setting up libgnome-keyring0:arm
Setting up libglade2-0:arm64 (1:
Setting up liborbit-2-0:arm64 (1
Setting up libbonobo2-0:arm64 (2
Processing triggers for mime-sup
Processing triggers for desktop-
Processing triggers for sgml-bas
Processing triggers for bamfdaem
Rebuilding /usr/share/applicatio
Processing triggers for libc-bin
Processing triggers for man-db (
Processing triggers for gnome-me
Setting up libgconf-2-4:arm64 (3
Setting up libgnomecanvas2-0:arm
Setting up gconf-service-backend
Setting up gconf-service (3.2.6-
Setting up gconf2 (3.2.6-4ubuntu
Setting up libgnomevfs2-common (
Setting up libgnomevfs2-0:arm64
Setting up libgnome2-common (2.3
Setting up libgnome-2-0:arm64 (2
Setting up libbonoboui2-0:arm64
Setting up libgnomeui-0:arm64 (2
Setting up camorama (0.19-5) ...
Processing triggers for libc-bin
aaksha@atharv-desktop:~$ camoram
Gtk-Message: 23:45:37.581: Faile
_IceTransSocketUNIXConnect: Cann
_IceTransSocketUNIXConnect: Cann

(camorama:14184): GnomeUI-WARNIN
Could not open network socket.
Opening in BLOCKING MODE

(camorama:14184): Gtk-WARNING **: 23:45:38.372: GtkSpinButton: setting an adjustment with non-zero page size is deprecated

(camorama:14184): GLib-GObject-WARNING **: 23:45:40.374: cannot retrieve class for invalid (unclassed) type '<invalid>'
```

Camorama - UVC Camera (046d:0825) - 640x480

Contrast: 34
Brightness: 216
Color: 138
Hue: 70
White Balance: 45

23.50 fps - current   21.43 fps - average

Figure 4: Camorama functionality on Jetson



```
Unpacking libgnomeui-0:arm64 (2.24.5-3.2) ...
Selecting previously unselected package camorama.
Preparing to unpack .../22-camorama_0.19-5_arm64.deb
Unpacking camorama (0.19-5) ...
Setting up libart-2.0-2:arm64 (2
Setting up gconf2-common (3.2.6-

Creating config file /etc/gconf/
Setting up libgnomeui-common (2.
Setting up libgnome-keyring-comm
Setting up libgnomecanvas2-commo
Setting up libbonobo2-common (2.
Setting up libbonoboui2-common (
Setting up libgnome-keyring0:arm
Setting up libglade2-0:arm64 (1:
Setting up liborbit-2-0:arm64 (1
Setting up libbonobo2-0:arm64 (2
Processing triggers for mime-sup
Processing triggers for desktop-
Processing triggers for sgml-bas
Processing triggers for bamfdaem
Rebuilding /usr/share/applicatio
Processing triggers for libc-bin
Processing triggers for man-db (
Processing triggers for gnome-me
Setting up libgconf-2-4:arm64 (3
Setting up libgnomecanvas2-0:arm
Setting up gconf-service-backend
Setting up gconf-service (3.2.6-
Setting up gconf2 (3.2.6-4ubuntu
Setting up libgnomevfs2-common (
Setting up libgnomevfs2-0:arm64
Setting up libgnome2-common (2.3
Setting up libgnome-2-0:arm64 (2
Setting up libbonoboui2-0:arm64
Setting up libgnomeui-0:arm64 (2
Setting up camorama (0.19-5) ...
Processing triggers for libc-bin
aaksha@atharv-desktop:~$ camoram
Gtk-Message: 23:45:37.581: Faile
_IceTransSocketUNIXConnect: Cann
_IceTransSocketUNIXConnect: Cann

(camorama:14184): GnomeUI-WARNIN
Could not open network socket.
Opening in BLOCKING MODE

(camorama:14184): Gtk-WARNING **: 23:45:38.372: GtkSpinButton: setting an adjustment with non-zero page size is deprecated

(camorama:14184): GLib-GObject-WARNING **: 23:45:40.374: cannot retrieve class for invalid (unclassed) type '<invalid>'
```

Camorama - UVC Camera (046d:0825) - 640x480

Contrast: 32
Brightness: 173
Color: 114
Hue: 70
White Balance: 45

30.50 fps - current    22.44 fps - average

Figure 5: Camorama features variation for obtaining desired features

3) [10 points] Using your verified Logitech C200 camera on a DE1-SoC, Raspberry Pi or Jetson, verify that it can stream continuously using to a raw image buffer for transformation and processing using example code from the computer-vision or computer_vision_cv3_tested folder such as simple-capture, simpler-capture, or simpler-capture-2. Read the code and modify the device that is opened if necessary to get this to work. Provide a screen shot to prove that you got continuous capture to work. Note that simpler capture requires installation of OpenCV on your DE1-SoC, Raspberry Pi, Jetson, or native Linux system. For the Jetson this will likely already be available on your board, but if not, please follow simple instructions found here to install openCV [the "Option 2, Building the public OpenCV library from source" is the recommended approach with –DWITH_CUDA=OFF. Don't install CUDA and please leave it off when you build OpenCV. For the DE1-SoC please find the files soc_system.rbf and SettingUp.pdf on Canvas. They contain instructions for setting up board and installing OpenCV. The TAs have set up using these and are able to complete exercise 4 requirements. The cmake command for opencv installation has been changed so that it works on the board too. Alternatively, you can use OpenCV port found here: http://rocketboards.org/foswiki/view/Projects/OpenCVPort. If you have trouble getting

OpenCV to work on your board, try running it on your laptop under Linux first. You can use OpenCV install, OpenCV with Python bindings for 2.x and 3.x for this.

Solution:

- ➔ Below screenshots are for simpler-capture and simpler-capture-2. The second screenshot shows a stream of images being stored in the disk.

a) Simpler Capture :

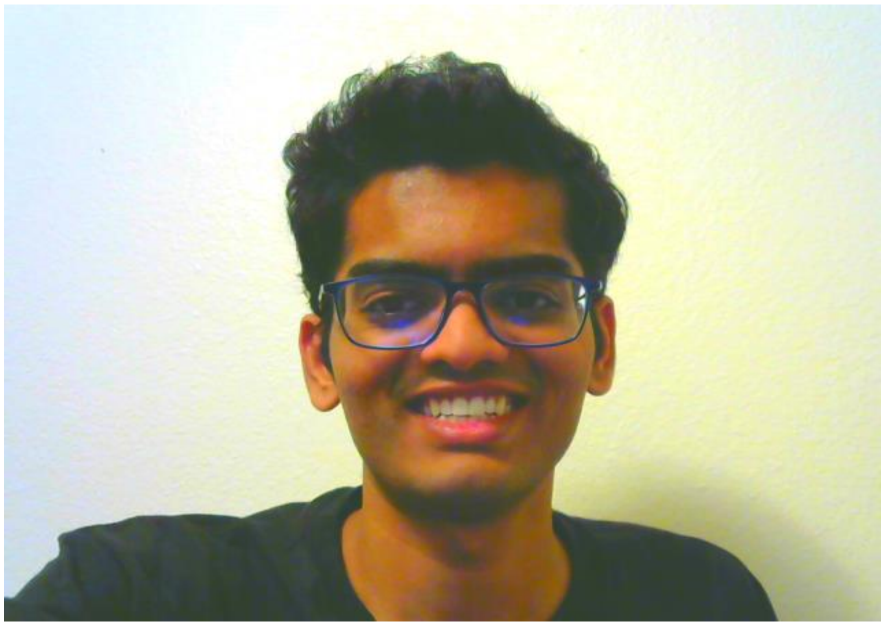
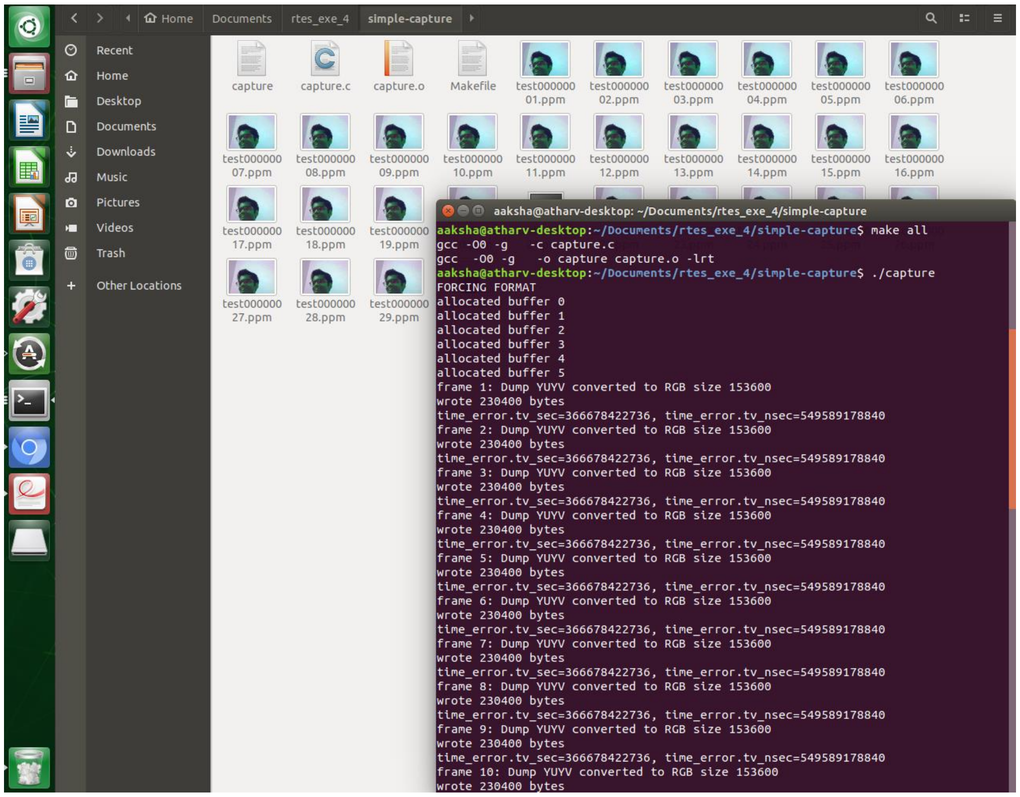


Figure 6: Simple Capture screenshot

- ➔ cvNamedWindow function in the code is used to create a window that is used as a placeholder for images as well as trackbars.
- ➔ cvImageShow function is implemented to display the image in this named window.
- ➔ cvDestroyWindow can be used to close the window and deallocate the used memory.
- ➔ When we execute ./capture, it allocates the buffers and then generates frames of images.

b) Simple-capture-2:



```
aaksha@atharv-desktop:~/Documents/rtes_exe_4/simple-capture$ make all
gcc -O0 -g   -c capture.c
gcc  -O0 -g   -o capture capture.o -lrt
aaksha@atharv-desktop:~/Documents/rtes_exe_4/simple-capture$ ./capture
FORCING FORMAT
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 153600
wrote 230400 bytes
time_error.tv_sec=366678422736, time_error.tv_nsec=549589178840
frame 2: Dump YUYV converted to RGB size 153600
wrote 230400 bytes
time_error.tv_sec=366678422736, time_error.tv_nsec=549589178840
frame 3: Dump YUYV converted to RGB size 153600
wrote 230400 bytes
time_error.tv_sec=366678422736, time_error.tv_nsec=549589178840
frame 4: Dump YUYV converted to RGB size 153600
wrote 230400 bytes
time_error.tv_sec=366678422736, time_error.tv_nsec=549589178840
frame 5: Dump YUYV converted to RGB size 153600
wrote 230400 bytes
time_error.tv_sec=366678422736, time_error.tv_nsec=549589178840
frame 6: Dump YUYV converted to RGB size 153600
wrote 230400 bytes
time_error.tv_sec=366678422736, time_error.tv_nsec=549589178840
frame 7: Dump YUYV converted to RGB size 153600
wrote 230400 bytes
time_error.tv_sec=366678422736, time_error.tv_nsec=549589178840
frame 8: Dump YUYV converted to RGB size 153600
wrote 230400 bytes
time_error.tv_sec=366678422736, time_error.tv_nsec=549589178840
frame 9: Dump YUYV converted to RGB size 153600
wrote 230400 bytes
time_error.tv_sec=366678422736, time_error.tv_nsec=549589178840
frame 10: Dump YUYV converted to RGB size 153600
wrote 230400 bytes
```

Figure 7: Stream of frames and no. of bytes written for them

- ➔ Here, a stream of images are being captured and as soon as the captured image is shown as NULL, then the algorithm stops capturing the images.
- ➔ cvSetCapture method is used to set the height and width of the image frames.
- ➔ cvNamedWindow is used to  create simpler capture 2 window

4) [20 points] Choose a continuous transformation OpenCV example from computer-vision such as the canny-interactive, hough-interactive, hough-eliptical-interactive, or stereotransform-impoved  or the from the same 4 transforms in computer vision cv3 tested.   Show a screen shot to prove you built and ran the code.  Provide a detailed explanation of the code and research uses for the continuous transformation by looking up API functions in the

OpenCV manual (http://docs.opencv.org ) and for stereo-transform-improved either implement or explain how you could make this work continuously rather than snapshot only.

Solution :

In this question, we have undertaken three transforms named as follows

a) Canny Interactive:

- ➔ The Canny edge detector detects a range of edges in images using a multi-stage algorithm.
- ➔ The Process of Canny edge detection algorithm can be broken down to 5 different steps:
  - i. Implementing Gaussian filter - smooths the image for noise cancellation
  - ii. Evaluating image intensity gradients.
  - iii. Intermediate suppression to avoid edges' spurious response.
  - iv. Double threshold - determine potential edges
  - v. Hysteresis to track edge.
- ➔ Following are the screenshots of image output for the executed canny interactive code.
- ➔ The canny interactive follows edge detection algorithm and there is an interactive bar in the window to set the threshold for the edges.
- ➔ The createTrackbar API is used to execute the canny algorithm and set the threshold for it and also the global variables are referenced using this function.
- ➔ cvCreateCameraCapture is used to capture image from the particular device which is by default set as 0.

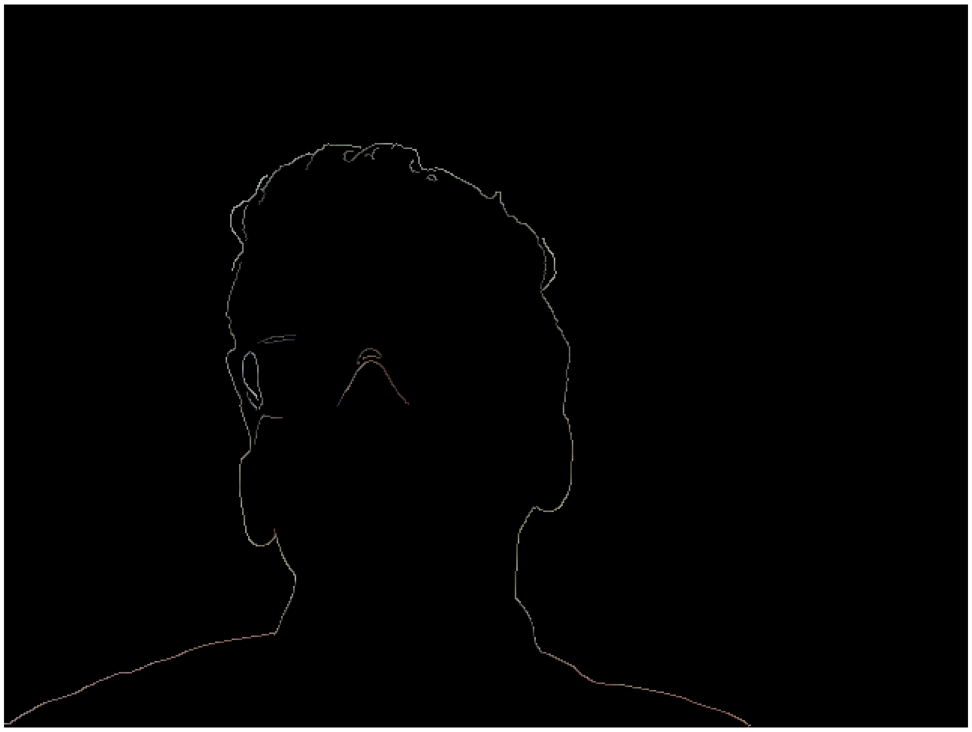


Figure 8: Simple Canny Interactive Execution

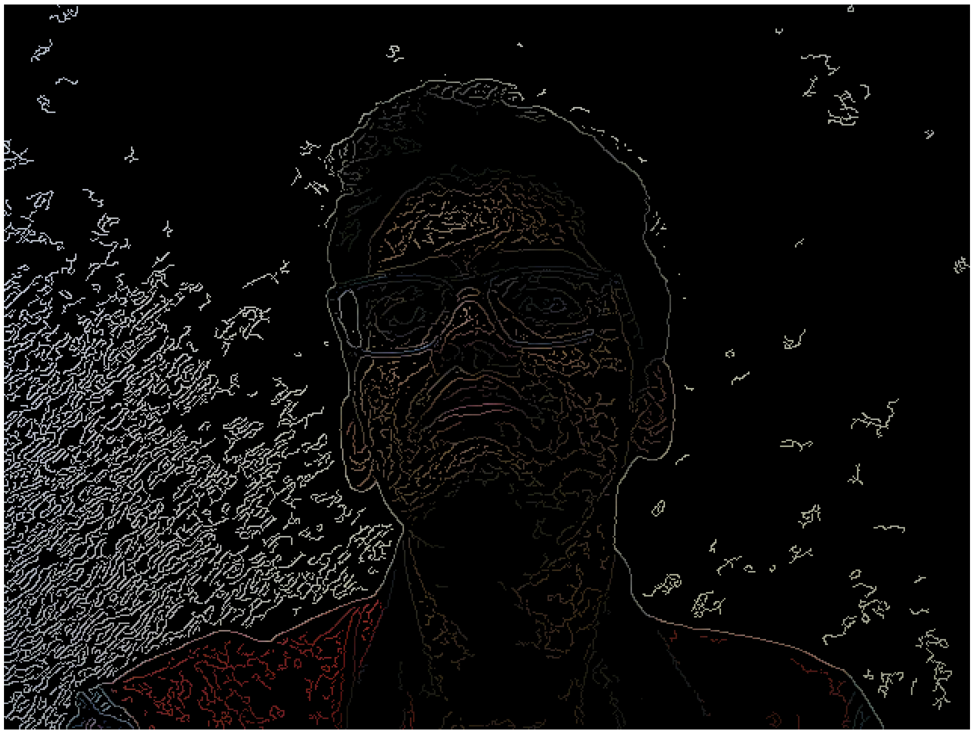
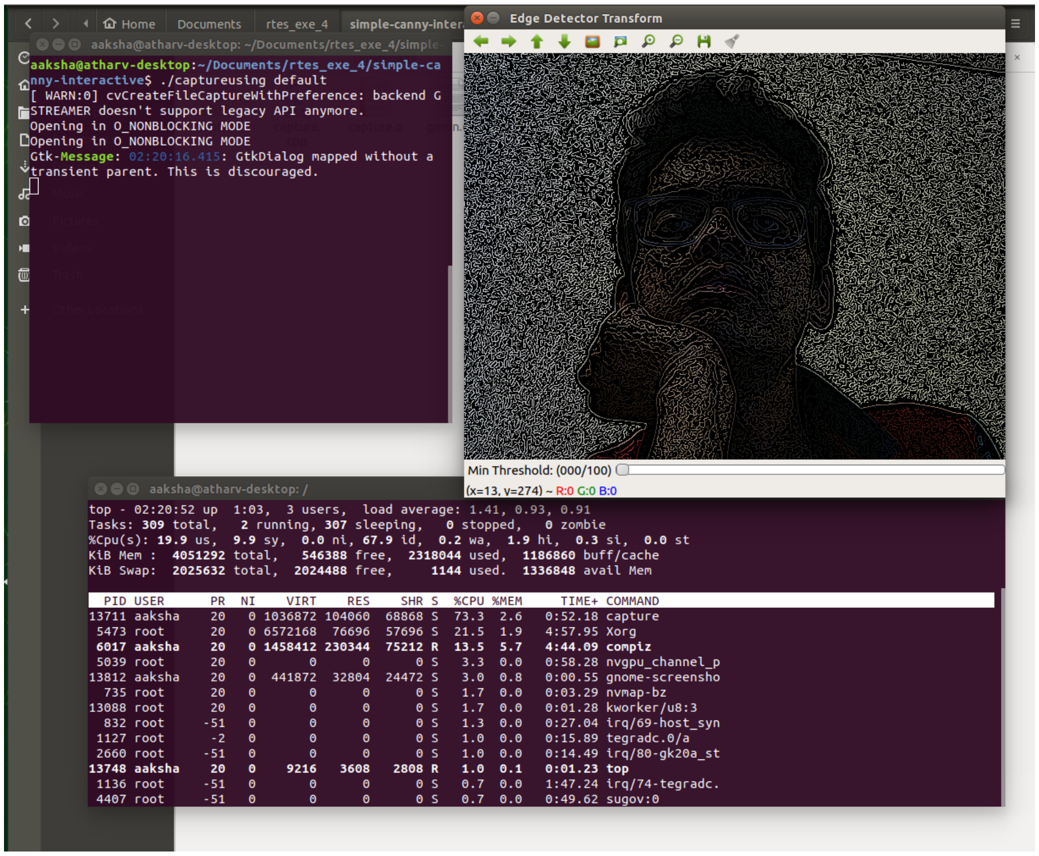


Figure 9: Simple Canny Interactive Execution with varied edge detection threshold



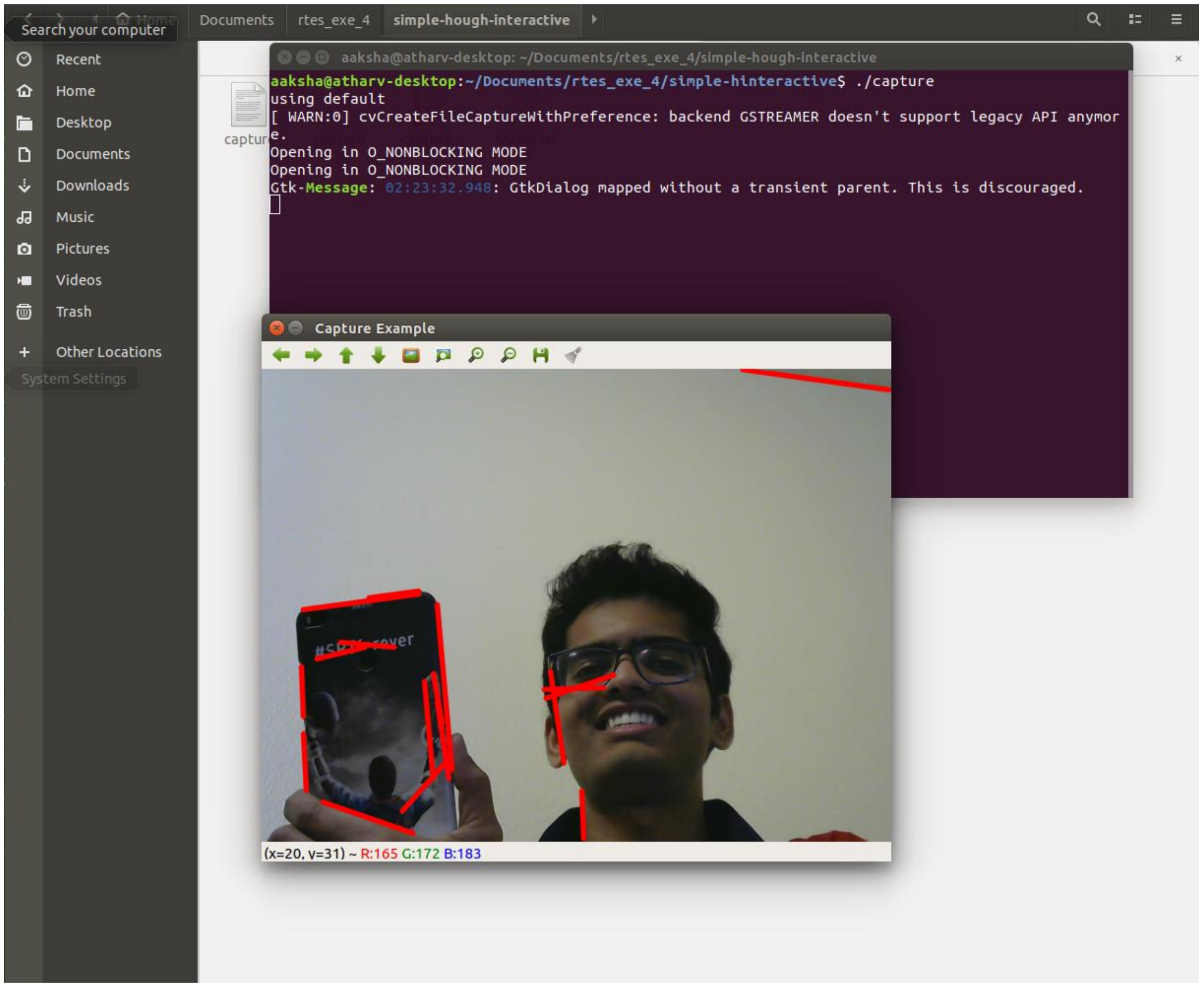
```
aaksha@atharv-desktop:~/Documents/rtes_exe_4/simple-ca
nny-interactive$ ./captureusing default
[ WARN:0] cvCreateFileCaptureWithPreference: backend G
STREAMER doesn't support legacy API anymore.
Opening in O_NONBLOCKING MODE
Opening in O_NONBLOCKING MODE
Gtk-Message: 02:20:16.415: GtkDialog mapped without a
transient parent. This is discouraged.
```

```
top - 02:20:52 up  1:03,  3 users,  load average: 1.41, 0.93, 0.91
Tasks: 309 total,   2 running, 307 sleeping,   0 stopped,   0 zombie
%Cpu(s): 19.9 us,  9.9 sy,  0.0 ni, 67.9 id,  0.2 wa,  1.9 hi,  0.3 si,  0.0 st
KiB Mem :  4051292 total,   546388 free,  2318044 used,  1186860 buff/cache
KiB Swap:  2025632 total,  2024488 free,     1144 used.  1336848 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
13711 aaksha    20   0 1036872 104060  68868 S  73.3  2.6   0:52.18 capture
 5473 root      20   0 6572168  76696  57696 S  21.5  1.9   4:57.95 Xorg
 6017 aaksha    20   0 1458412 230344  75212 R  13.5  5.7   4:44.09 compiz
 5039 root      20   0       0      0      0 S   3.3  0.0   0:58.28 nvgpu_channel_p
13812 aaksha    20   0  441872  32804  24472 S   3.0  0.8   0:00.55 gnome-screensho
  735 root      20   0       0      0      0 S   1.7  0.0   0:03.29 nvmap-bz
13088 root      20   0       0      0      0 S   1.7  0.0   0:01.28 kworker/u8:3
  832 root     -51   0       0      0      0 S   1.3  0.0   0:27.04 irq/69-host_syn
 1127 root      -2   0       0      0      0 S   1.0  0.0   0:15.89 tegradc.0/a
 2660 root     -51   0       0      0      0 S   1.0  0.0   0:14.49 irq/80-gk20a_st
13748 aaksha    20   0    9216   3608   2808 R   1.0  0.1   0:01.23 top
 1136 root     -51   0       0      0      0 S   0.7  0.0   1:47.24 irq/74-tegradc.
 4407 root     -51   0       0      0      0 S   0.7  0.0   0:49.62 sugov:0
```

Figure 10: Simple Canny Interactive Execution CPU Loading with 73.3 %

b) <u>Simple Hough Interactive Transform</u> :

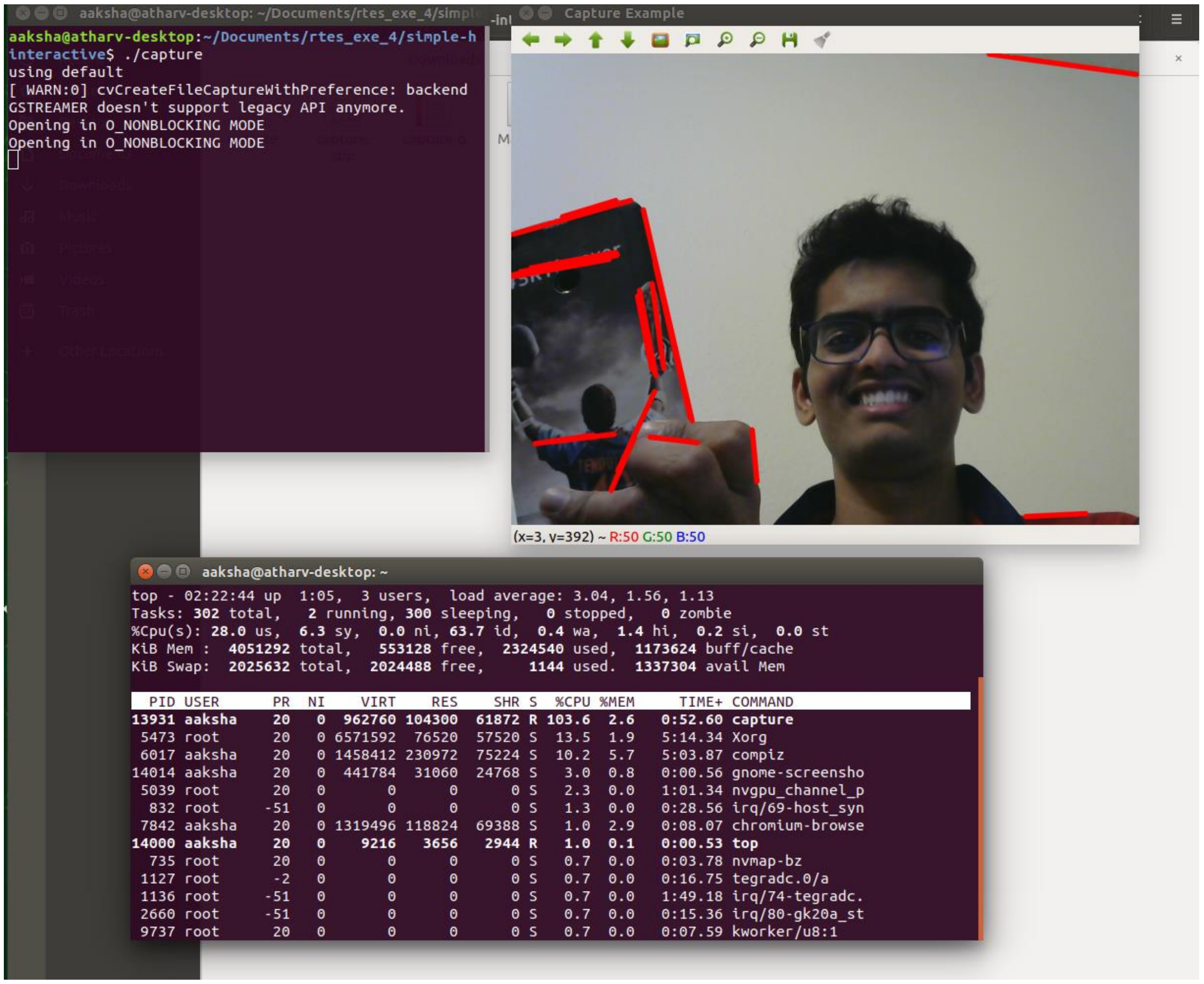
➔ The Hough transform technique is implemented for isolating the particular shapes within an image.
➔ The Hough transform can be used to identify the parameter(s) of a curve which best fits a set of given edge points.
➔ It can be useful in many places like calculating the number of trunks in plant trees since plant leaves can make it difficult to give the head count.
➔ Also, it can be used to detect the wall edges in the blurred images due to smoke.
➔ In the code, HoughLinesP are used to detect the lines and mat_frames function is also included to initialize the image window.



```
aaksha@atharv-desktop:~/Documents/rtes_exe_4/simple-hinteractive$ ./capture
using default
[ WARN:0] cvCreateFileCaptureWithPreference: backend GSTREAMER doesn't support legacy API anymor
e.
Opening in O_NONBLOCKING MODE
Opening in O_NONBLOCKING MODE
Gtk-Message: 02:23:32.948: GtkDialog mapped without a transient parent. This is discouraged.
```

Figure 11: Simple Hough Interactive Execution with line detection

- ➔ Simple Hough Interactive transform takes comparatively higher CPU Loading than the Simple Canny Interactive.
- ➔ Simple Hough interactive transform requires CPU loading of 103.6%



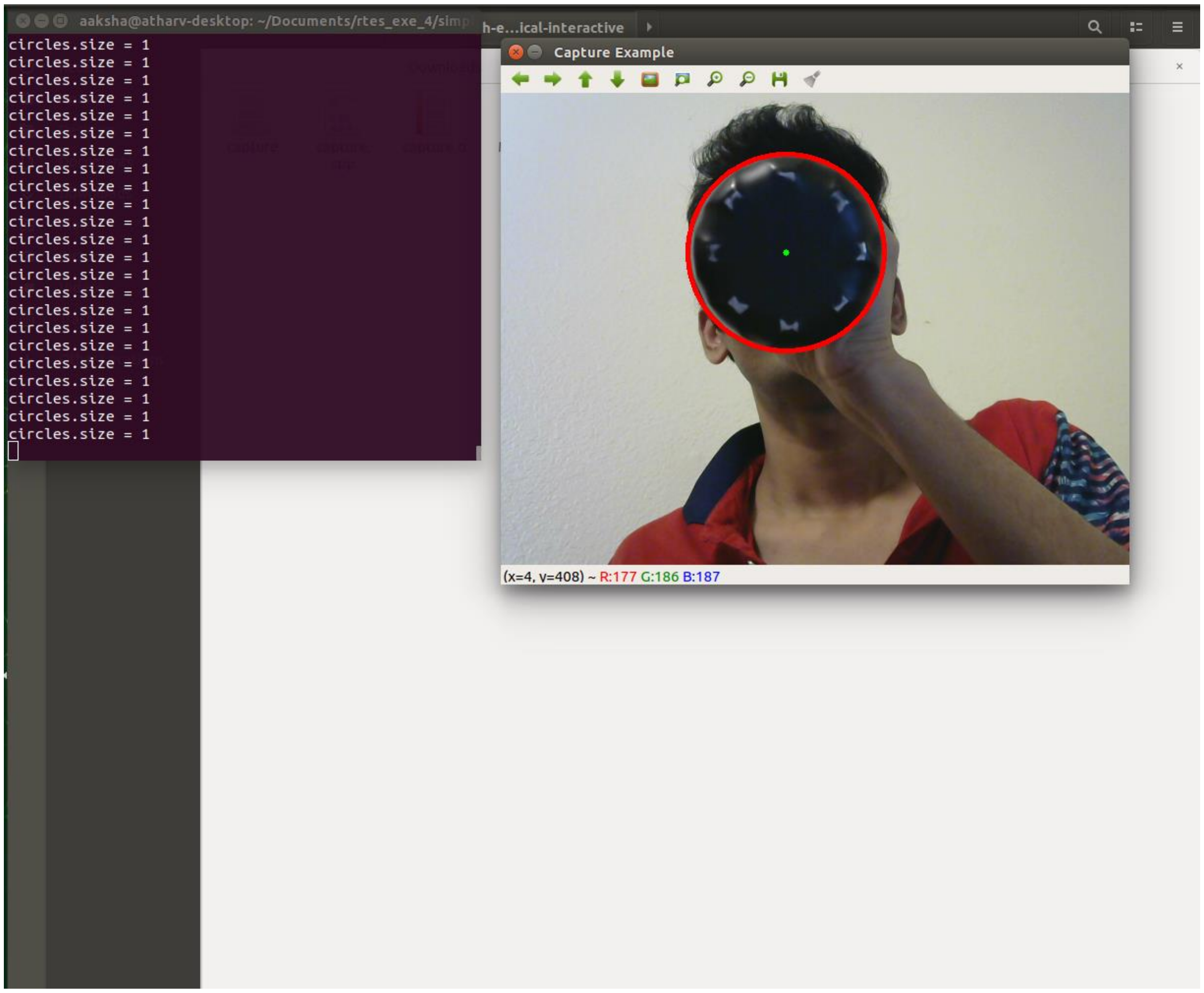
```
aaksha@atharv-desktop:~/Documents/rtes_exe_4/simple-h
interactive$ ./capture
using default
[ WARN:0] cvCreateFileCaptureWithPreference: backend
GSTREAMER doesn't support legacy API anymore.
Opening in O_NONBLOCKING MODE
Opening in O_NONBLOCKING MODE
```

```
top - 02:22:44 up  1:05,  3 users,  load average: 3.04, 1.56, 1.13
Tasks: 302 total,   2 running, 300 sleeping,   0 stopped,   0 zombie
%Cpu(s): 28.0 us,  6.3 sy,  0.0 ni, 63.7 id,  0.4 wa,  1.4 hi,  0.2 si,  0.0 st
KiB Mem :  4051292 total,   553128 free,  2324540 used,  1173624 buff/cache
KiB Swap:  2025632 total,  2024488 free,     1144 used.  1337304 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
13931 aaksha    20   0  962760 104300  61872 R 103.6  2.6   0:52.60 capture
 5473 root      20   0 6571592  76520  57520 S  13.5  1.9   5:14.34 Xorg
 6017 aaksha    20   0 1458412 230972  75224 S  10.2  5.7   5:03.87 compiz
14014 aaksha    20   0  441784  31060  24768 S   3.0  0.8   0:00.56 gnome-screensho
 5039 root      20   0       0      0      0 S   2.3  0.0   1:01.34 nvgpu_channel_p
  832 root     -51   0       0      0      0 S   1.3  0.0   0:28.56 irq/69-host_syn
 7842 aaksha    20   0 1319496 118824  69388 S   1.0  2.9   0:08.07 chromium-browse
14000 aaksha    20   0    9216   3656   2944 R   1.0  0.1   0:00.53 top
  735 root      20   0       0      0      0 S   0.7  0.0   0:03.78 nvmap-bz
 1127 root      -2   0       0      0      0 S   0.7  0.0   0:16.75 tegradc.0/a
 1136 root     -51   0       0      0      0 S   0.7  0.0   1:49.18 irq/74-tegradc.
 2660 root     -51   0       0      0      0 S   0.7  0.0   0:15.36 irq/80-gk20a_st
 9737 root      20   0       0      0      0 S   0.7  0.0   0:07.59 kworker/u8:1
```

Figure 12: Simple Hough Interactive Execution CPU loading

c) <u>Hough Eliptical Transform</u>

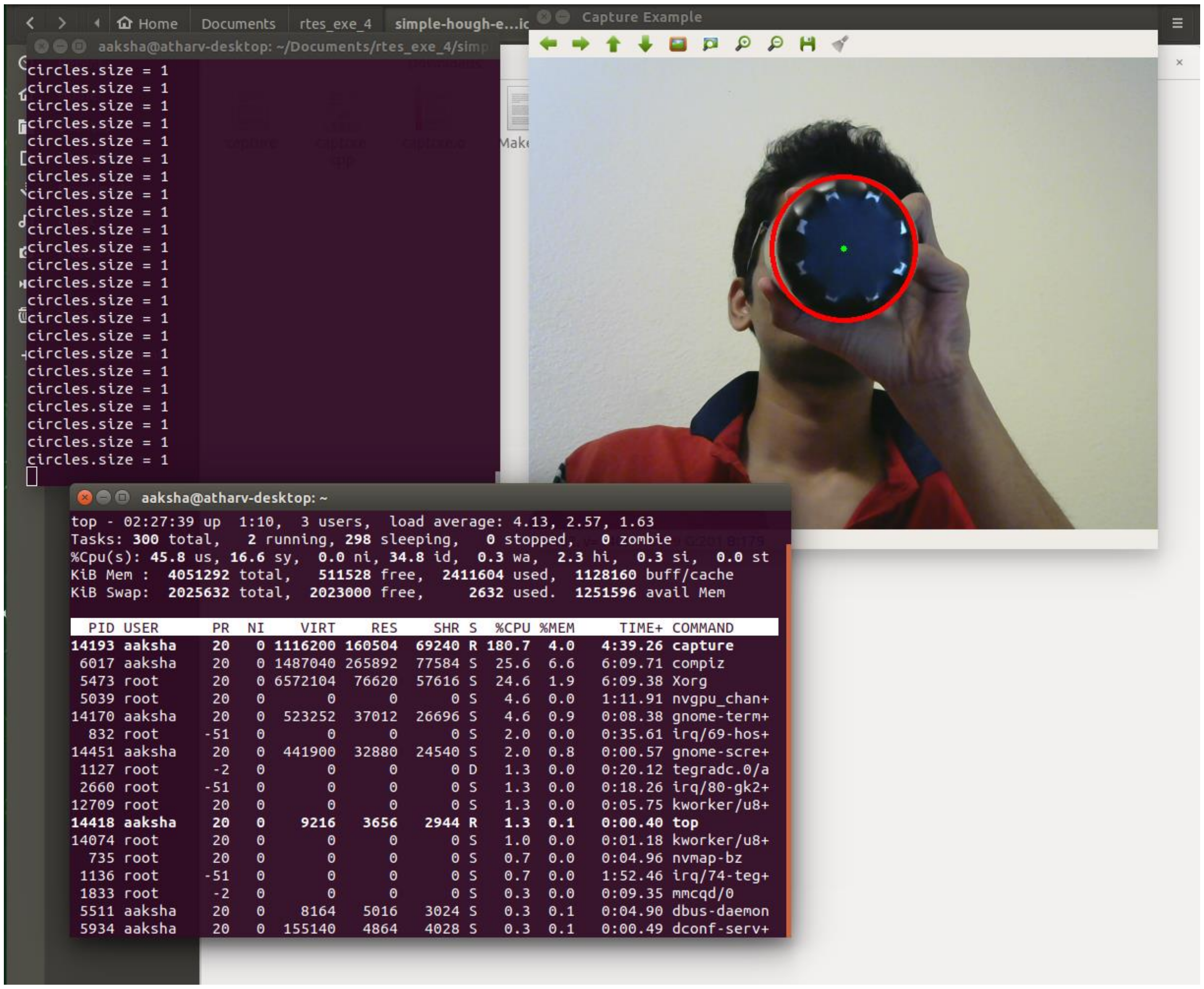
- ➔ Shown below is the hough elliptical transform with circles detected in the image along with their size



```
aaksha@atharv-desktop: ~/Documents/rtes_exe_4/simp
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
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
circles.size = 1
circles.size = 1
circles.size = 1
```

Figure 13: Hough Eliptical Execution with circle detection

➔ Hough Eliptical Transform consumes the maximum CPU power of 180.6% compared to simple canny interactive and simple hough interactive transform.



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

aaksha@atharv-desktop: ~

```
top - 02:27:39 up  1:10,  3 users,  load average: 4.13, 2.57, 1.63
Tasks: 300 total,   2 running, 298 sleeping,   0 stopped,   0 zombie
%Cpu(s): 45.8 us, 16.6 sy,  0.0 ni, 34.8 id,  0.3 wa,  2.3 hi,  0.3 si,  0.0 st
KiB Mem :  4051292 total,   511528 free,  2411604 used,  1128160 buff/cache
KiB Swap:  2025632 total,  2023000 free,     2632 used.  1251596 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
14193 aaksha    20   0 1116200 160504  69240 R 180.7  4.0   4:39.26 capture
 6017 aaksha    20   0 1487040 265892  77584 S  25.6  6.6   6:09.71 compiz
 5473 root      20   0 6572104  76620  57616 S  24.6  1.9   6:09.38 Xorg
 5039 root      20   0       0      0      0 S   4.6  0.0   1:11.91 nvgpu_chan+
14170 aaksha    20   0  523252  37012  26696 S   4.6  0.9   0:08.38 gnome-term+
  832 root     -51   0       0      0      0 S   2.0  0.0   0:35.61 irq/69-hos+
14451 aaksha    20   0  441900  32880  24540 S   2.0  0.8   0:00.57 gnome-scre+
 1127 root      -2   0       0      0      0 D   1.3  0.0   0:20.12 tegradc.0/a
 2660 root     -51   0       0      0      0 S   1.3  0.0   0:18.26 irq/80-gk2+
12709 root      20   0       0      0      0 S   1.3  0.0   0:05.75 kworker/u8+
14418 aaksha    20   0    9216   3656   2944 R   1.3  0.1   0:00.40 top
14074 root      20   0       0      0      0 S   1.0  0.0   0:01.18 kworker/u8+
  735 root      20   0       0      0      0 S   0.7  0.0   0:04.96 nvmap-bz
 1136 root     -51   0       0      0      0 S   0.7  0.0   1:52.46 irq/74-teg+
 1833 root      -2   0       0      0      0 S   0.3  0.0   0:09.35 mmcqd/0
 5511 aaksha    20   0    8164   5016   3024 S   0.3  0.1   0:04.90 dbus-daemon
 5934 aaksha    20   0  155140   4864   4028 S   0.3  0.1   0:00.49 dconf-serv+
```

Figure 14: Simple Hough Eliptical Execution CPU loading

5) [50 points] Using a Logitech C200, choose 3 real-time interactive transformations to compare in terms of average frame rate at a given resolution for a range of at least 3 resolutions in a common aspect ratio (e.g. 4:3 for 1280x960, 640x480, 320x240, 160x120, 80x60) by adding time-stamps (there should be a logging thread) to analyze the potential throughput. You should get at least 9 separate datasets running 1 transformation at time. Based on average analysis, pick a reasonable soft real-time deadline (e.g. if average frame rate is 12 Hz, choose a deadline of 100 milliseconds to provide some margin) and convert the processing to SCHED_FIFO and determine if you can meet deadlines with predictability and measure statistical jitter in the frame rate relative to your deadline for both the default scheduler and SCHED_FIFO in each case.

<u>Solution:</u>

## Design Concepts

- ➔ The code uses the first half of the data set for calculating the deadline. The deadline is determined by calculating the maximum execution time and adding an extra 20% to increase the margin.
- ➔ The rest of the samples of the data set are used for jitter analysis. Jitter is calculated relative to the deadline.
- ➔ The execution time is calculated by using clock_gettime function for executing the transform
- ➔ The data for jitter analysis is dumped into a CSV file using file io which is then used to plot a graph.
- ➔ The code has 5 macros to control the execution, VRES HRES for changing resolution and 3 others for applying different transforms. The code has one thread that applies the selected transform through macros.

## Algorithm and Prototype Analysis

The following table shows average frame rate for different resolutions for different transforms over 100 samples

| Transform | 1280x960 | 640x480 | 320x240 | 800x600 | 960x720 |
|---|---|---|---|---|---|
| Canny | 9 fps | 18 fps | 21 fps | 17 fps | 12 fps |
| Hough | 3 fps | 5 fps | 17 fps | 3 fps | 4 fps |
| Hough Elliptical | 2 fps | 14 fps | 19 fps | 5 fps | 4 fps |

The following table shows the dynamically determined deadline over 50 samples for different transforms at different resolutions.

| Transform | 1280x960 | 640x480 | 320x240 | 800x600 | 960x720 |
|---|---|---|---|---|---|
| Canny | 115 ms | 67 ms | 67 ms | 67 ms | 89 ms |
| Hough | 271 ms | 203 ms | 65 ms | 235 ms | 300 ms |
| Hough Elliptical | 356 ms | 82 ms | 67 ms | 217 ms | 237 ms |

The following table shows the average jitter over 50 samples for different transforms at different resolutions. The average jitter is calculated for FIFO scheduling.

| Transform | 1280x960 | 640x480 | 320x240 | 800x600 | 960x720 |
|---|---|---|---|---|---|
| Canny | 19.21 | 12.28 | 12.08 | 12.22 | 12.38 |
| Hough | 52.87 | 35.34 | 11.73 | 37.13 | 38.57 |
| Hough Elliptical | 73.78 | 15.22 | 12.763 | 43.90 | 47.53 |

The following table shows missed deadlines for FIFO scheduling.

| Transform | 1280x960 | 640x480 | 320x240 | 800x600 | 960x720 |
|---|---|---|---|---|---|
| Canny | 0 | 0 | 0 | 0 | 0 |
| Hough | 0 | 0 | 0 | 0 | 0 |
| Hough Elliptical | 0 | 0 | 0 | 0 | 0 |

The following table shows missed deadline CFS scheduling

| Transform | 1280x960 | 640x480 | 320x240 | 800x600 | 960x720 |
|---|---|---|---|---|---|
| Canny | 1 | 3 | 0 | 1 | 0 |
| Hough | 0 | 0 | 0 | 0 | 0 |
| Hough Elliptical | 29 | 2 | 2 | 4 | 19 |

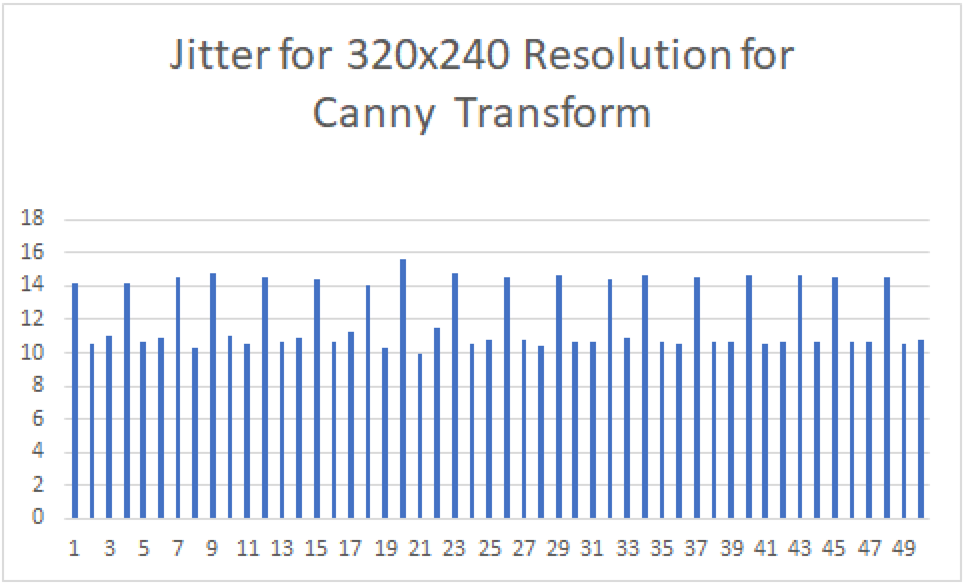
Analysis from above Results

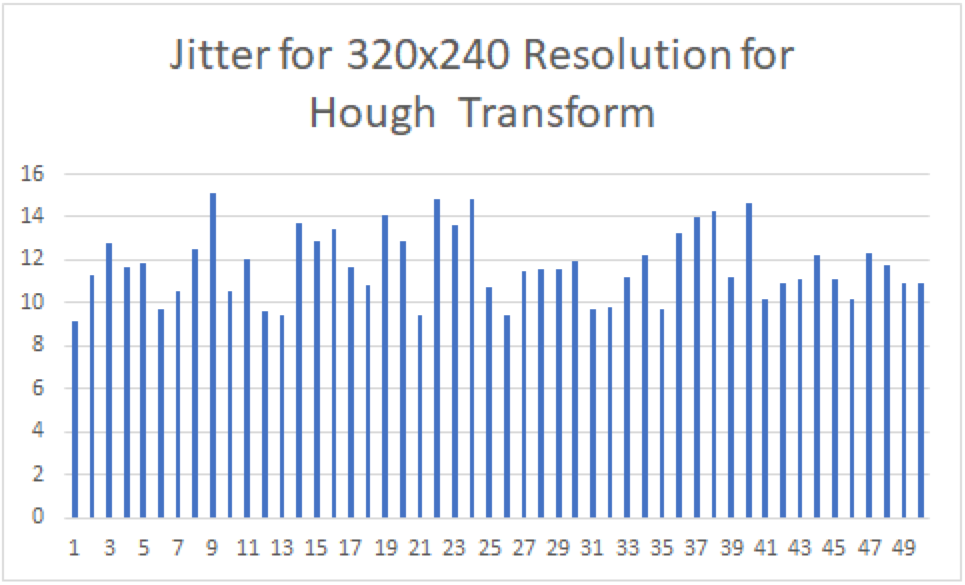
- ➔ The frame rate increases as the resolution decrease. The transforms act on each pixel so as the resolution decreases the number of frames processed will increase.
- ➔ The deadline is increasing with the increase in resolution as more time would be needed to process the frames.
- ➔ The Average jitter is calculated with FIFO Scheduling policy so it is always positive and increases with an increase in resolution as no deadline is missed with FIFO policy.
- ➔ No deadlines are missed when the FIFO scheduler is used so it is indeed best of the two schedulers used. It can be used for Hard real-time systems as meeting the deadlines is guaranteed.

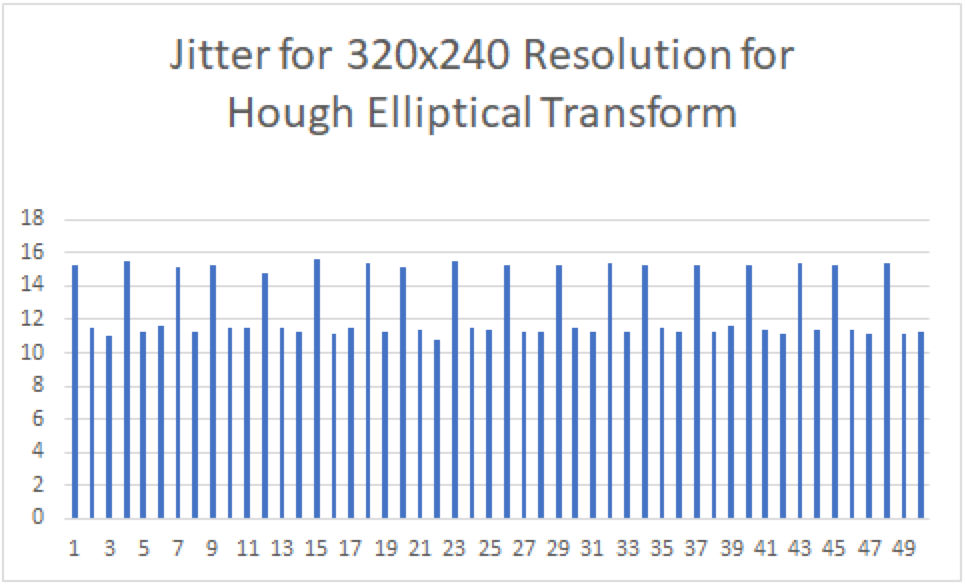
- ➔ CFS is not as useful here because it is missing deadlines in some cases. It is particularly proven worse with the Hough elliptical transform as it misses the most deadlines at all resolutions. Though canny transform performs a bit better at lower resolution and hough does not miss any deadlines. One can conclude that completely fair scheduler is unreliable and should not be used for a hard real-time system design

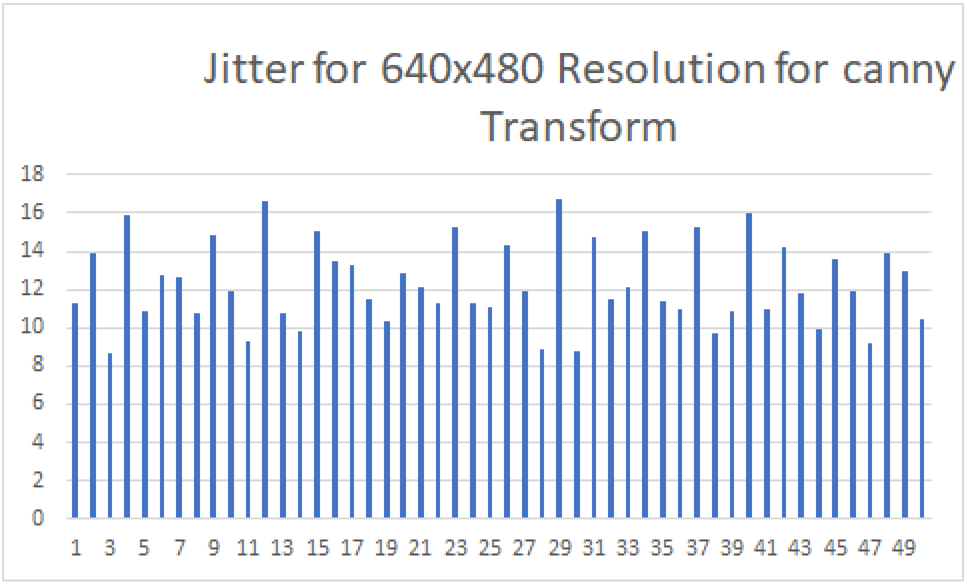
Jitter analysis

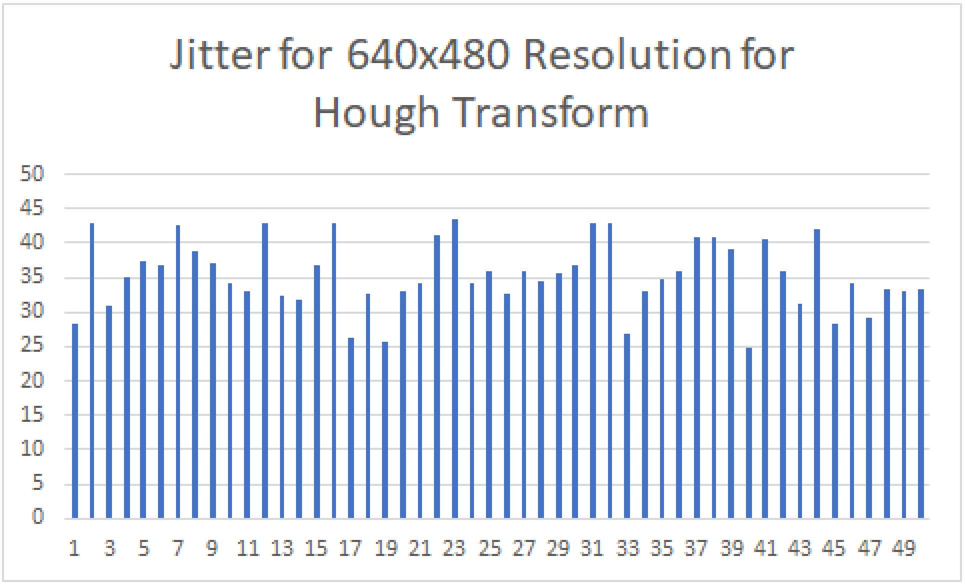
- ➔ The jitter values are dumped into a CSV file using file operations and they are used to plot a graph.
- ➔ The jitter is an undesirable quality of any signal and with stable timer services jitter should be reduced to increase the stability.
- ➔ In this particular case, the jitter is positive so no deadlines are missed hence even it is undesirable it is not showing a negative impact on the system.
- ➔ If the CFS scheduler is used then we would have seen negative jitter which would mean that deadlines are missed and that would have affected the system enough to render it useless for real-time purposes.
- ➔ The graphs are plotted for jitter in every case as shown below.

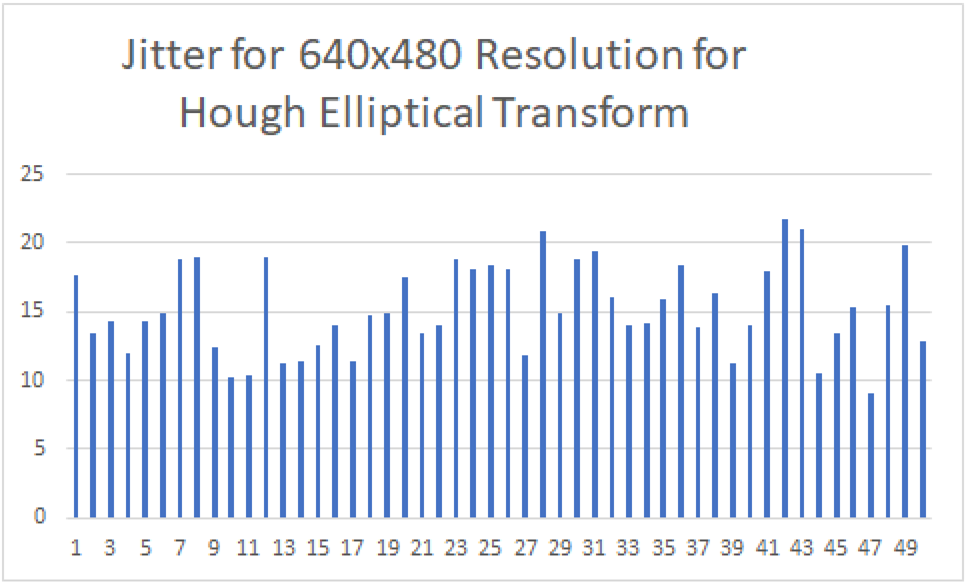

Jitter for 320x240 Resolution for Canny Transform
18
16
14
12
10
8
6
4
2
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49

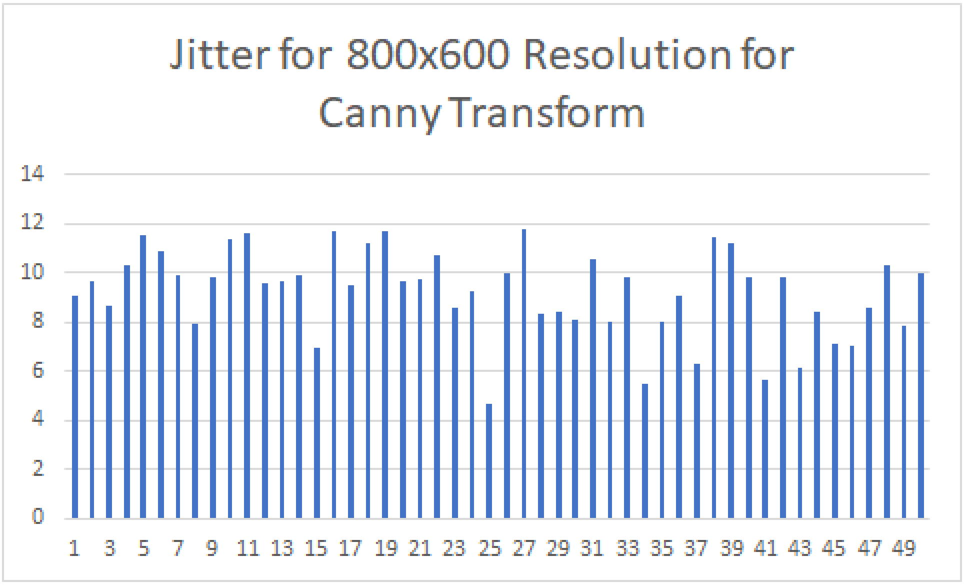

Jitter for 320x240 Resolution for Hough Transform
16
14
12
10
8
6
4
2
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49

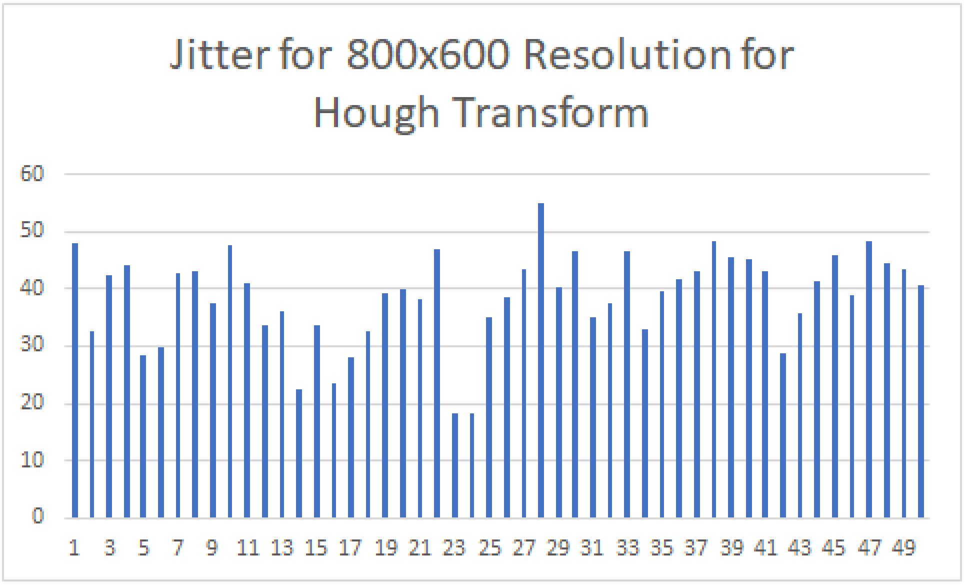

Jitter for 320x240 Resolution for Hough Elliptical Transform
18
16
14
12
10
8
6
4
2
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49

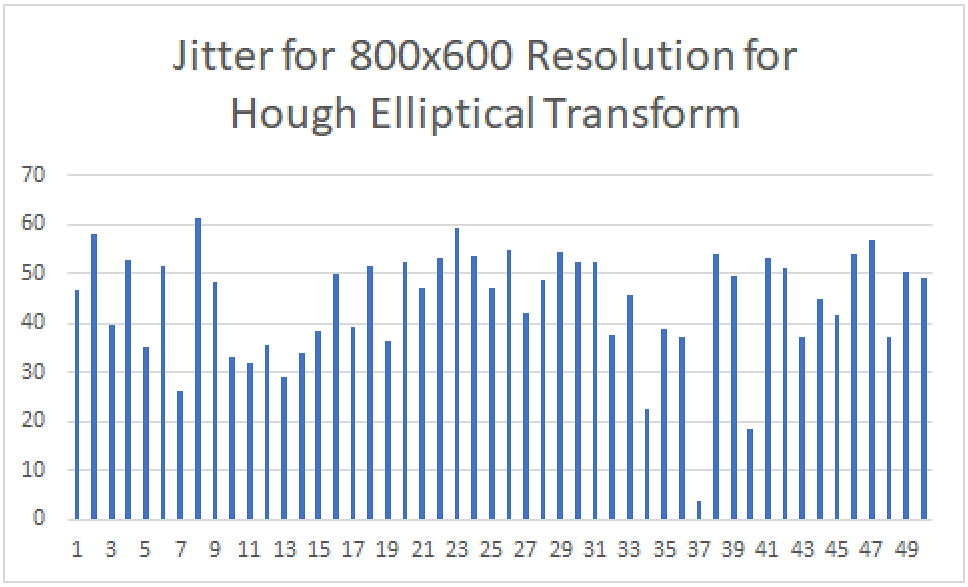

Jitter for 640x480 Resolution for canny Transform
18
16
14
12
10
8
6
4
2
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49


Jitter for 640x480 Resolution for Hough Transform
50
45
40
35
30
25
20
15
10
5
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49

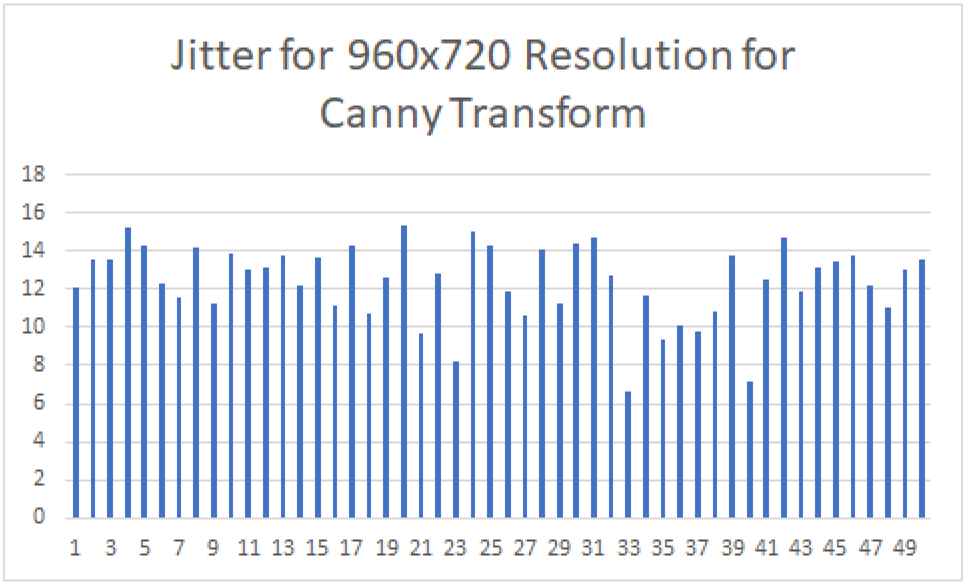

Jitter for 640x480 Resolution for Hough Elliptical Transform
25
20
15
10
5
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49


Jitter for 800x600 Resolution for Canny Transform
14
12
10
8
6
4
2
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49

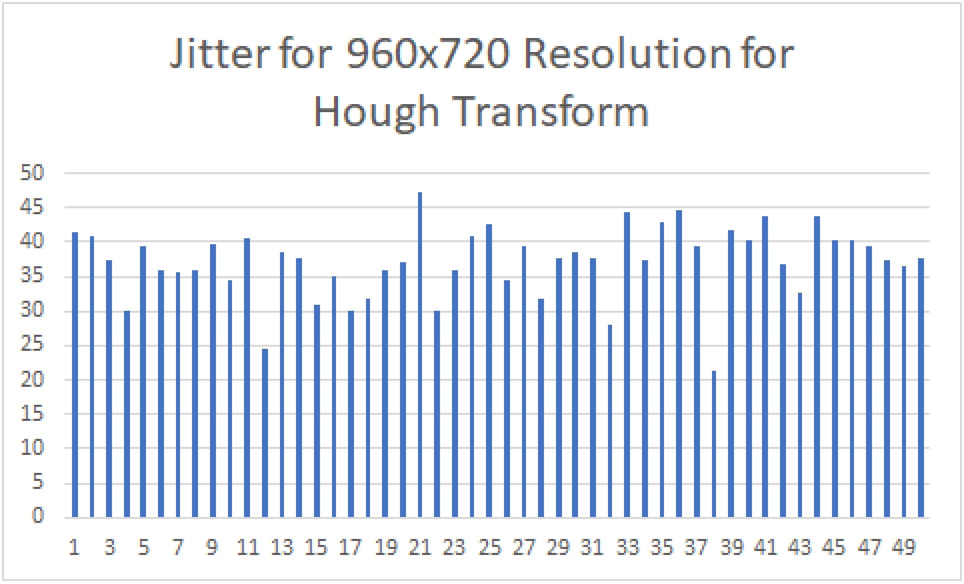

Jitter for 800x600 Resolution for Hough Transform
60
50
40
30
20
10
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49

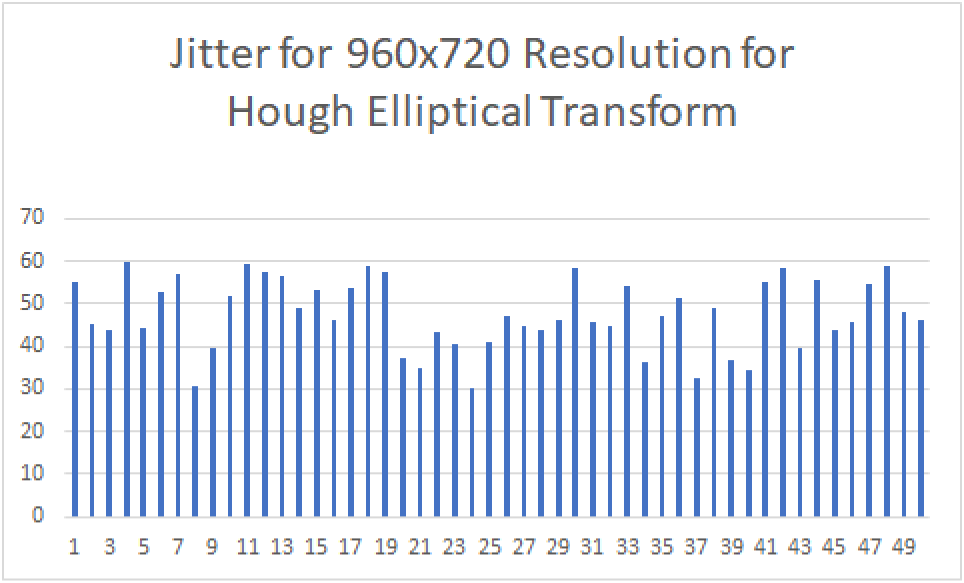


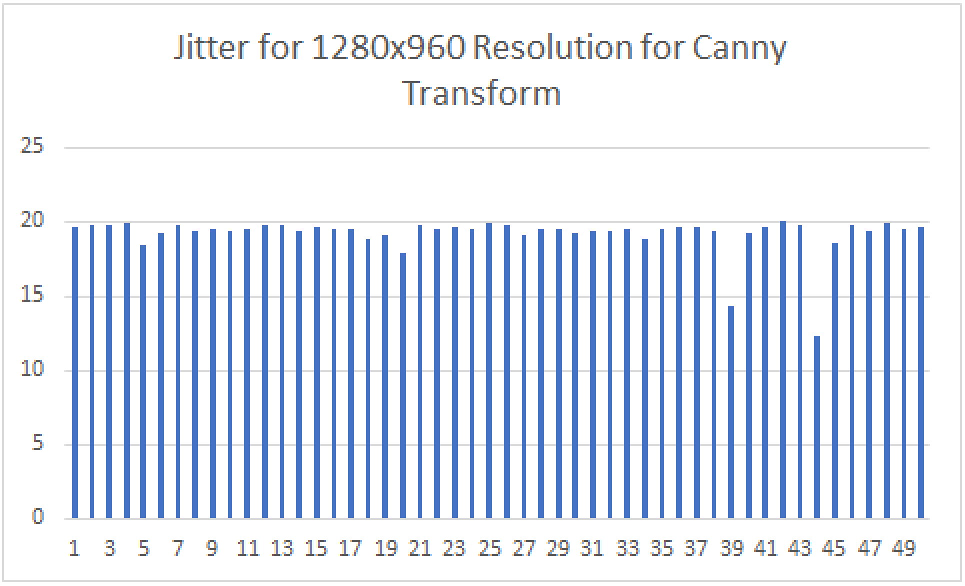
Jitter for 800x600 Resolution for Hough Elliptical Transform
70
60
50
40
30
20
10
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49

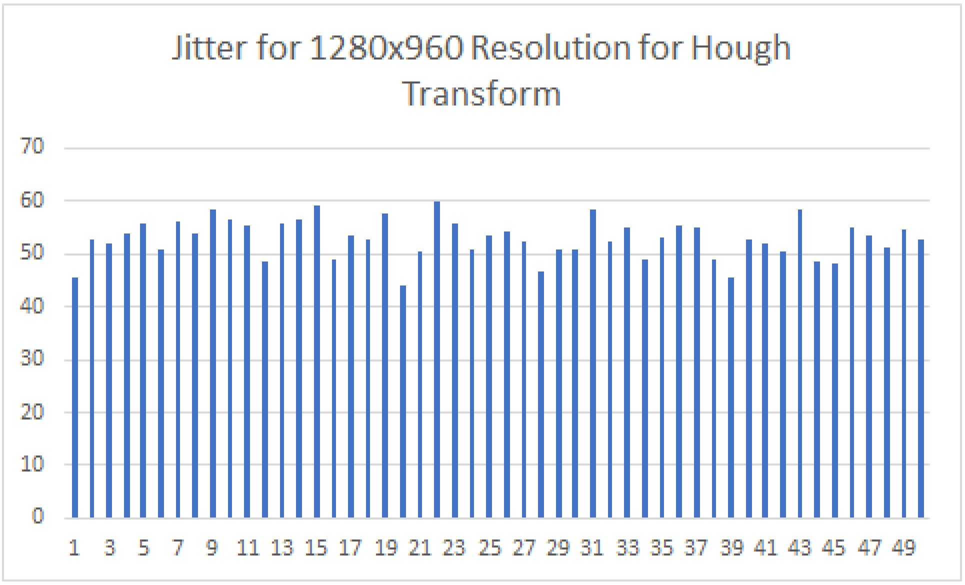


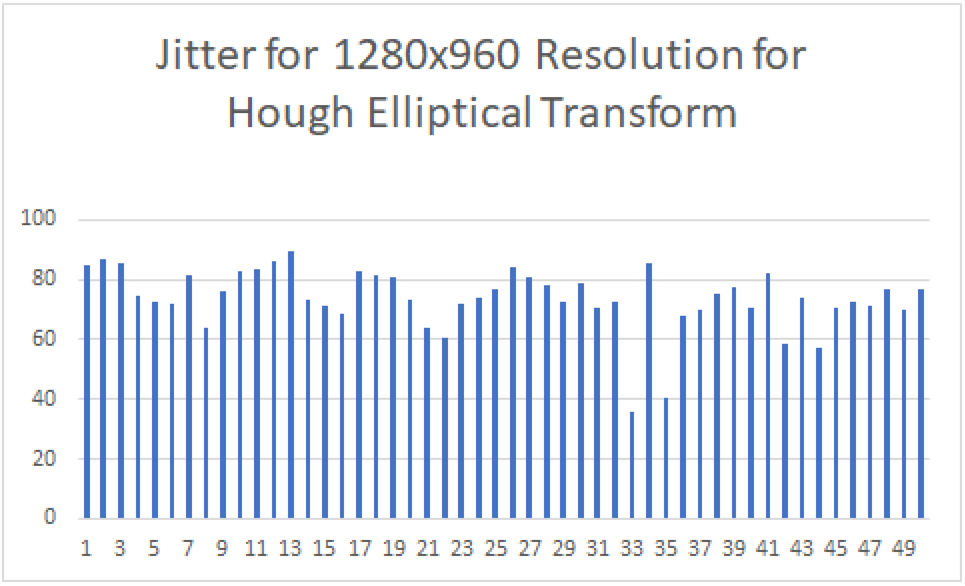
Jitter for 960x720 Resolution for Canny Transform
18
16
14
12
10
8
6
4
2
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49


Jitter for 960x720 Resolution for Hough Transform
50
45
40
35
30
25
20
15
10
5
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49


Jitter for 960x720 Resolution for Hough Elliptical Transform
70
60
50
40
30
20
10
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49


Jitter for 1280x960 Resolution for Canny Transform
25
20
15
10
5
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49


Jitter for 1280x960 Resolution for Hough Transform
70
60
50
40
30
20
10
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49



References:

1) https://docs.opencv.org/2.4/modules/highgui/doc/user_interface.html
2) https://en.wikipedia.org/wiki/Canny_edge_detector
3) https://homepages.inf.ed.ac.uk/rbf/HIPR2/hough.htm

Note:

1. Question 5 is done on my roommate's board instead of my partner's dues current COVID-19 situation as her board was available easily.
2. Question 1-4 are done on Jetson Nano shared with my Exercise 1 and final project partner Aaksha Jaywant.