809T Homework 3

The assignment is to read the images from the picam and detect the green light from the traffic signal. We also write the time taken by the raspberry pi and the camera to read and find the contours on the image. The video recorded is done in realtime and the link to video is given in the below link.
https://youtu.be/twHax2ms5rw

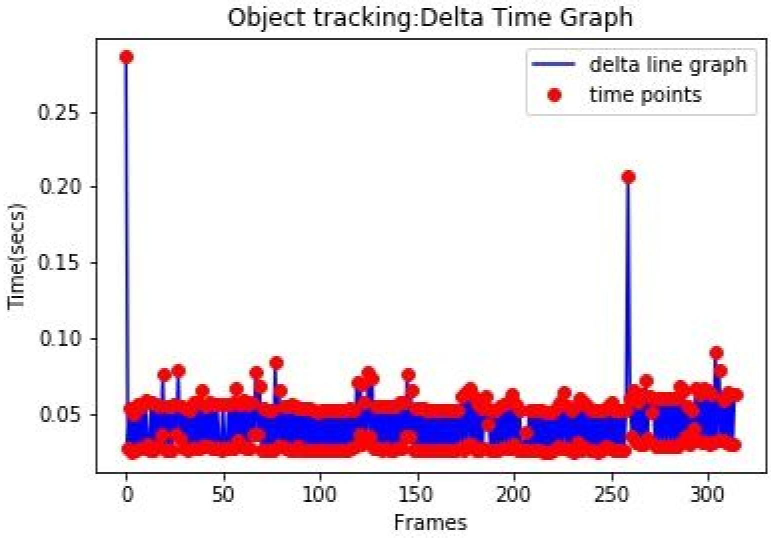
Code:
The code to read the frames and process it is shown below.

```
from picamera.array import PiRGBArray
from picamera import PiCamera
import time
import cv2
import datetime
import numpy as np
import imutils

camera = PiCamera()
camera.resolution = (640,480)
camera.framerate = 60
rawCapture = PiRGBArray(camera,size=(640,480))
fourcc = cv2.VideoWriter_fourcc(*'XVID')
out = cv2.VideoWriter("video_out.avi",fourcc,10,(640,480))
start_time = time.time()
file = open("hw3_data.txt","w+")
time_thresh = 175
for frame in camera.capture_continuous(rawCapture,format="bgr",use_video_port=False):
    start_time_ = datetime.datetime.now()
    image = frame.array
    lower_green  = np.array([40,200,100])
    upper_green = np.array([90,255,190])
    #Color Conversion and Mask conversion
    hsv = cv2.cvtColor(image,cv2.COLOR_BGR2HSV)
    mask= cv2.inRange(hsv,lower_green,upper_green)
    result=cv2.bitwise_and(hsv, hsv, mask=mask)
    #finding the countors in the image
    cnts  = cv2.findContours(mask.copy(),
cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)
    cnts = imutils.grab_contours(cnts)
    center = None
    #Check for the number of contours and draw the circle for the green signal
    if len(cnts) >0:
#       print("the cnts is",cnts)
        c = max(cnts,key = cv2.contourArea)
        #finds the x,y and the radius of the enclosed circle
        ((x, y), radius) = cv2.minEnclosingCircle(c)
        M = cv2.moments(c)
        #plots the circle if the radius is more than 1
        if radius > 1:
            cv2.circle(image, (int(x), int(y)), int(radius),(0, 125, 255), 2)
    #Displays the frame
    cv2.imshow("frame",image)
    key = cv2.waitKey(1) & 0xFF
    rawCapture.truncate(0)
    if key== ord("q"):
        break
    time_out = time.time()-start_time
    if time_out >= time_thresh:
        break

    stop_time = datetime.datetime.now()
    now = stop_time-start_time_
    out_values = str(now.total_seconds())+"\n"
    file.write(out_values)
    print("The time out is",time_out)
    out.write(image)
```

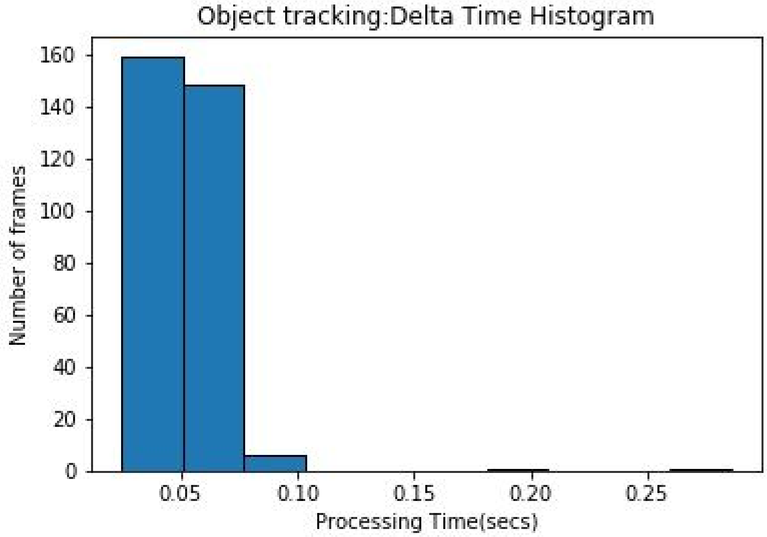
The contours drawn on the image is made into a circle since the signal is red. This can be used to detect any shape by changing the draw circle to drawcontours in python. The text file is written and is also plotted as follows.

Graph:

The graphs and the histogram plots are shown below.



The above graph shows the time taken by each frame against the time taken to process it.



The above graph shows the histogram of the time taken to the number of frames. The code to read and plot the frames are as follow.

```
import numpy as np
import matplotlib.pyplot as plt
#Reads the delta time from text value
delta_time = np.genfromtxt("hw3_data.txt")
plt.plot(delta_time,"b-",label = "delta line graph")
plt.plot(delta_time,"ro",label = "time points")
plt.legend()
plt.xlabel("Frames")
plt.ylabel("Time(secs)")
plt.title("Object tracking:Delta Time Graph")
plt.savefig("out1.jpg")
plt.show()
#plot for the histogram
plt.hist(delta_time,ec = "black")
plt.xlabel("Processing Time(secs)")
plt.ylabel("Number of frames")
plt.title("Object tracking:Delta Time Histogram")
plt.savefig("out2.jpg")
plt.show()
```