University of Colorado
Boulder

# EXERCISE 1

BY

SIDDHANT JAJOO

**UNDER THE GUIDANCE OF:**

PROFESSOR SAM SIEWERT,
UNIVERSITY OF COLORADO BOULDER

6/14/2019

**USED LINUX UBUNTU 18.04**

# TABLE OF CONTENTS

# QUESTION 1

## SUMMARY OF EXPLORE VIDEO ANALYTICS IN THE CLOUD

There are different tools to encode and decode digital video that have been widely used in day to day life unknowingly. MPEG, OpenCv and GNU Image Processing (GIMP) are some of the examples. CV applications involve acquisition, digital image formats for pixels (picture elements that represent points of illumination), images and sequences of them (movies), processing and transformation, segmentation, recognition, and ultimately scene descriptions.The use of Computer Vision (CV) to understand video content, either in real time or from precaptured databases of image sequences, is typically referred to as video analytics.

Every Image that needs to be processed or that is needed to extract some kind of information must undergo video analytics. Video Analytics is the basis of Computer and Machine vision related applications. Video Analytics is broadly defined as analysis of digital videos content from cameras or stored sequence of images. It involves several disciplines such as Image acquisition and encoding, CV, Machine Vision, Image Processing, Machine Learning, Real time and Interactive Systems, Storage, Networking, Database and Computing.

The basic structure of Video Analytics applications can be achieved in two ways:

1. Capturing of data through embedded intelligent sensors and processing of data i.e analytics both can be done on the same embedded device.
2. Breaking the above architecture in two segments: embedded intelligent sensors and cloud based video analytics system.

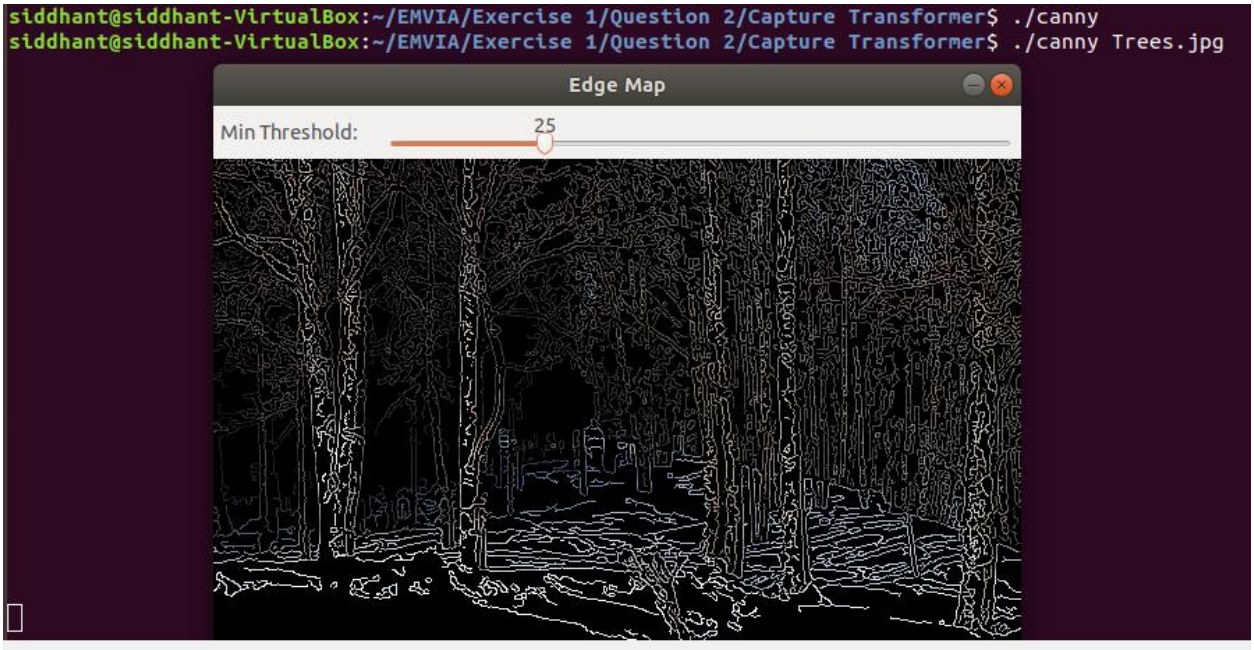
Now, embedding CV in smartphones and tablets may not always be practical due to resource constraints. So data can be relayed to data-centric systems such as data centers. This would enable for data processing on database created from data all over the world. With sufficient cloud-computing power, this can be achieved. This makes the segmentation approach to be more scalable and realizable as compared to having all the processing and computing power on the resource constraint device. The challenge that is faced in segmenting the architecture is that the processing power should be so high that the processes should be real time.A lot of applications can be thought using video analytics and machine vision which could serve of great purpose in the near future. Some can be developed at this point of time but some do require years of hard work. Some of the intuitive applications include Augmented Reality (AR), skeletal transformations to track movement, fully autonomous car, reliable face detection including expression feedback, virtual shopping, two way television, sign language interaction etc. These applications can be realized with the help of CV and video analytics.

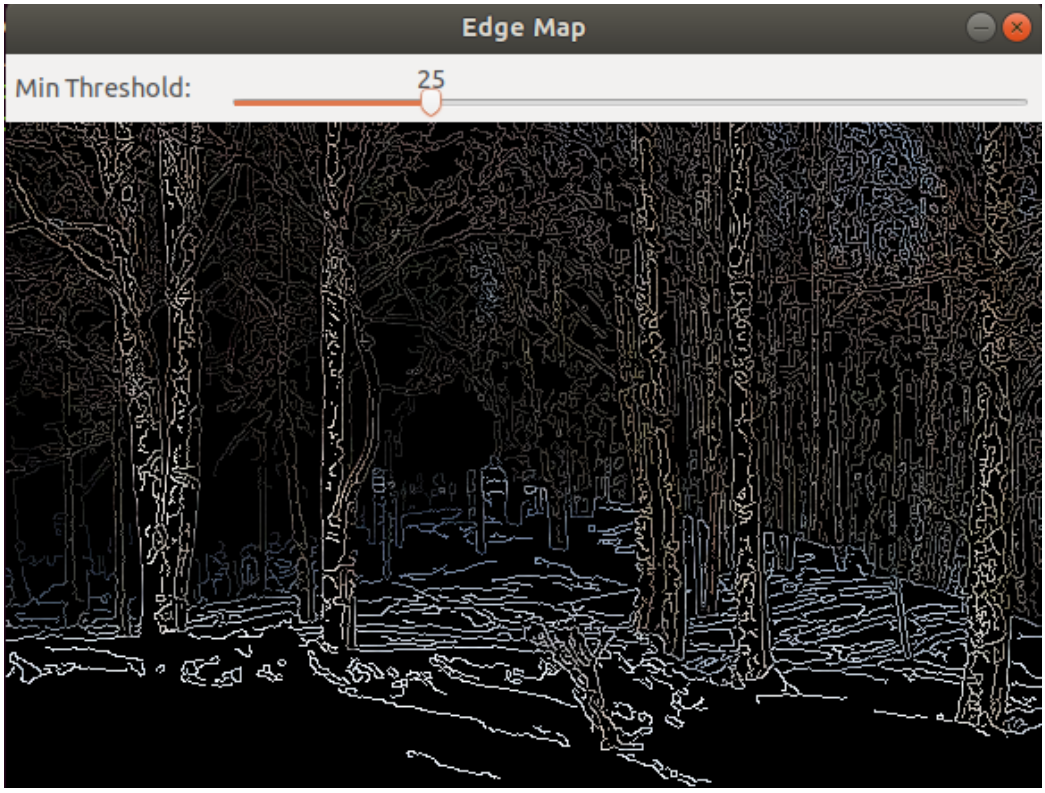
It can be concluded that video analytics has limitless future scope combined with computer vision especially in sign language and vision applications. It also shows a promising future in entertainment, social networking, telemedicine and medical industries. Image processing, machine learning and data centric computer architecture as mentioned above will play a crucial part in bringing together CV and video analytics in order to provide safety and products/services to consumers in everyday life.

# QUESTION 2

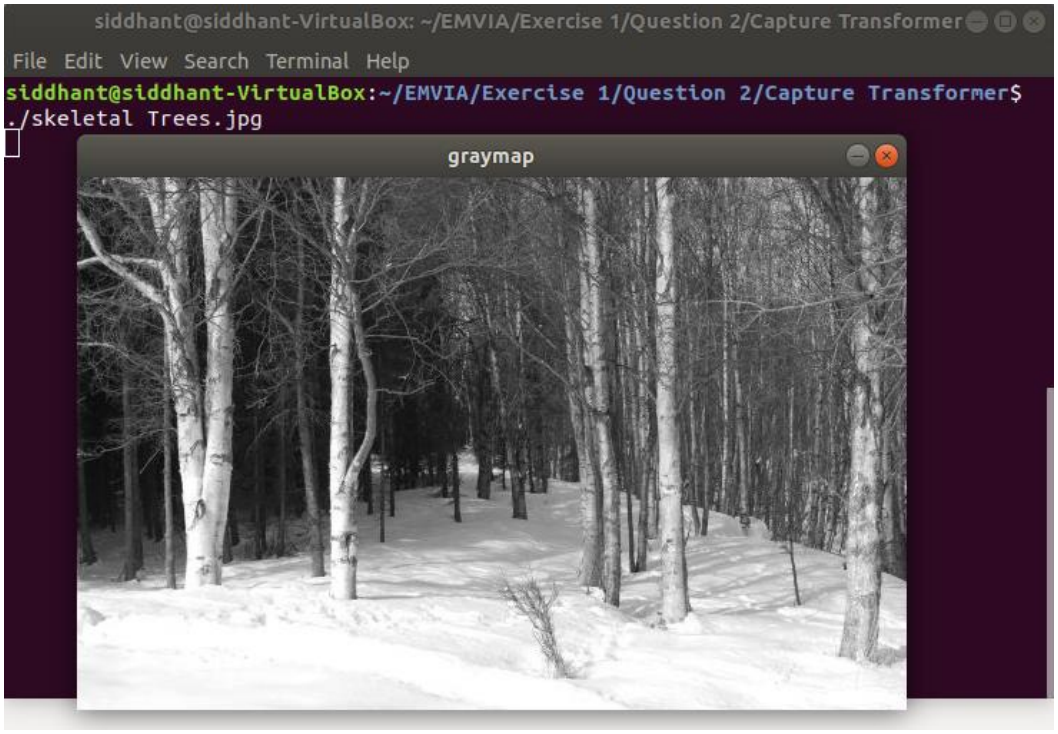
## ORIGINAL IMAGE

## CANNY TRANSFORM IMAGE





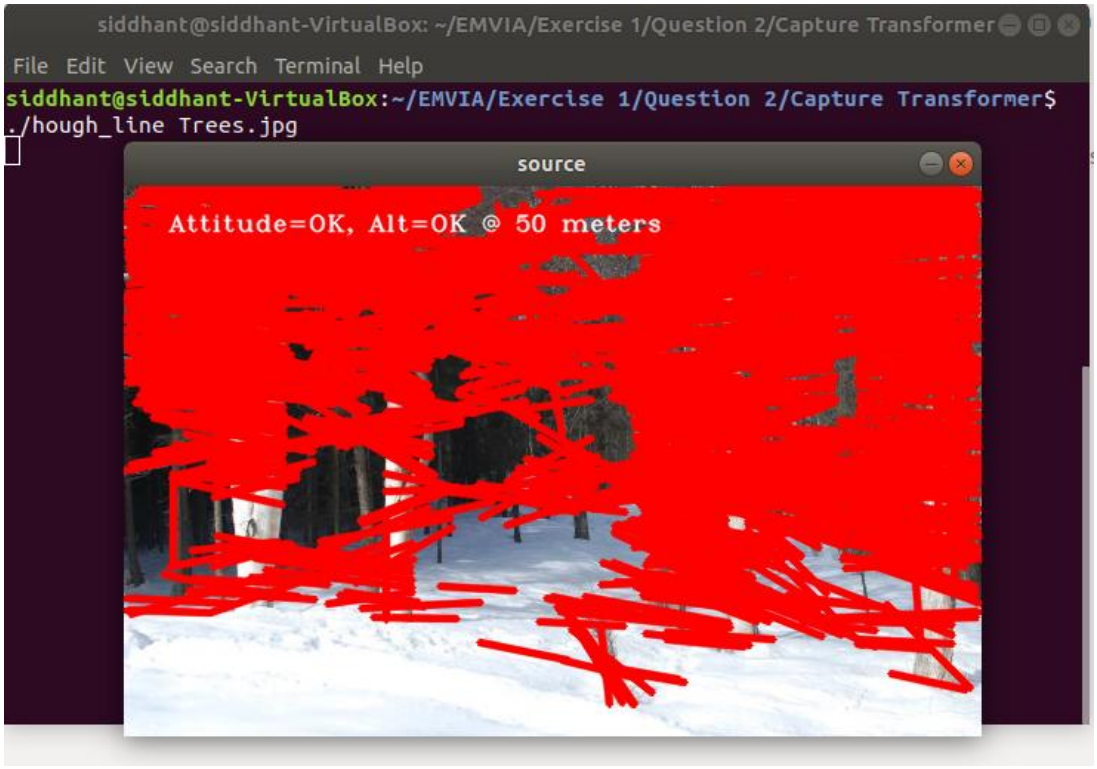

## SKELETAL TRANSFORM IMAGE

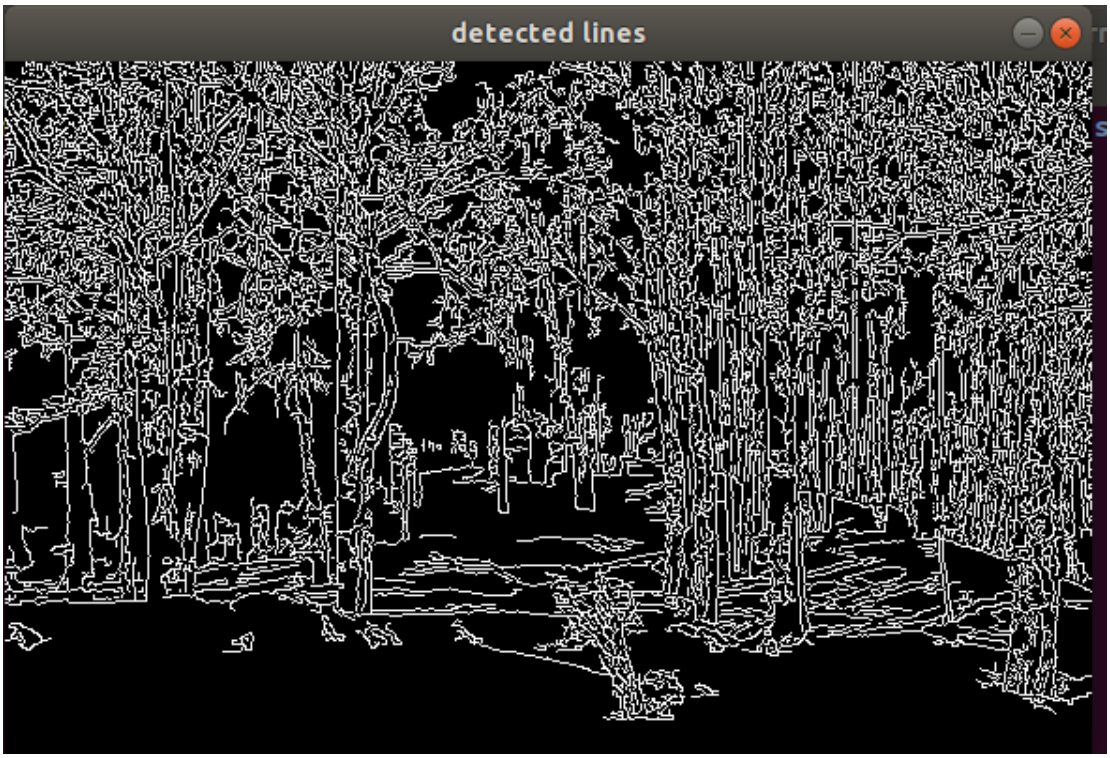



binary


skeleton

## HOUGH LINE TRANSFORM IMAGE



## SOBEL TRANSFORM IMAGE


## SOBEL TRANSFORM OPERATION

The Sobel Transform is an edge detection algorithm which is simply used to find regions in image where there is a steep change in pixel value or intensity value. A high value represents steep change and a low value represents shallow change. These changes are expressed in terms of derivatives. A high change in gradient represents a major change in the image. So, in order to detect an edge, individual pixel gradients are calculated to compare with the neighboring pixels. In this process an image is undergone two derivative calculations: the horizontal and vertical derivatives. Horizontal and Vertical changes are computed independently to detect a horizontal or vertical edge. The sobel transform is applied to each and every individual pixel and the corresponding values are calculated in the x and y direction. These values are then averaged using a formula to approximate the gradient from both the directions.

The formulas to calculate the horizontal changes, vertical changes and the average are as follows:

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * I \qquad G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} * I \qquad G = \sqrt{G_x^2 + G_y^2}$$

Horizontal changes | Vertical changes | Average

I = Image, Gx/Gy = Output, The matrix is an example kernel/filter.

The average equation gives us the gradient of the pixels. Higher the gradient higher is the change and thus a presence of an edge.

The kernel/filter isa matrix which is used to operate on the Image matrix. These kernels have different values depending upon their functionality. For example, the averaging/blur filter has a different value than the filter required for sobel transform in x and y directions. Perhaps, the filter for both x and y directions in sobel transform are different.

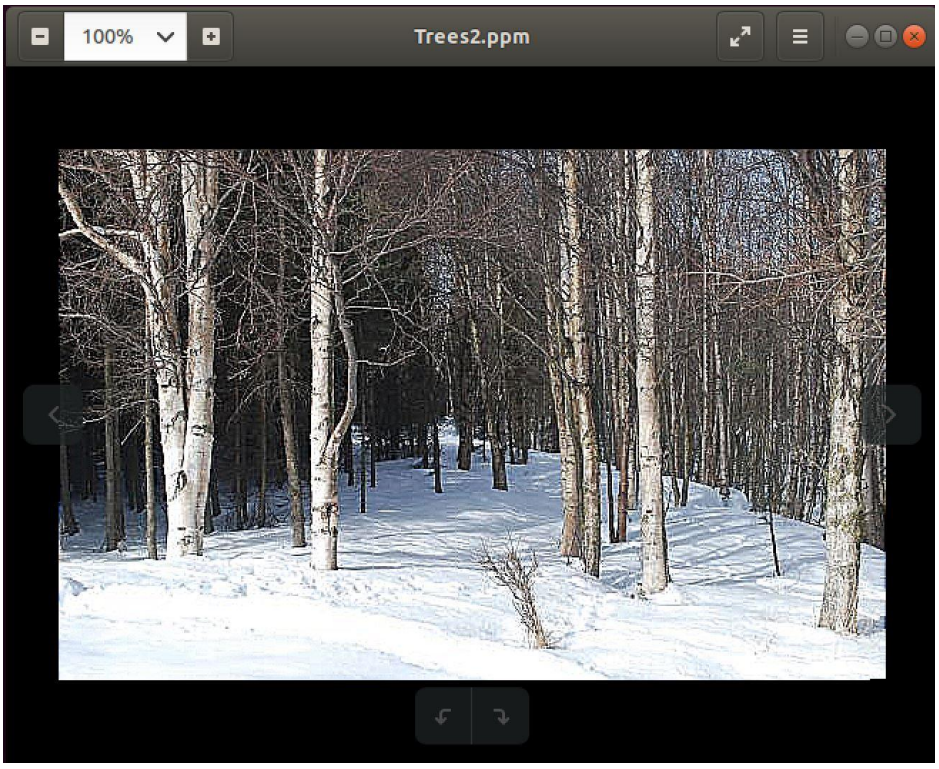
Steps to apply Sobel operator:

1. Firstly, a Gaussian Blur is applied to our image to reduce the noise in the Image. This is done to prevent the high frequency elements in the image to be detected as edges. Applying aGaussianblur would eliminate all the high frequency components and the image will only consists of solid edges.
2. The Image obtained in the first step is converted to a grayscale image.
3. Now, the derivatives are calculated in the x and y directions.
4. Finally, an approximate value is obtained by adding the gradients in x and y directions.
5. The final image is then displayed.

# QUESTION 3

## ORIGINAL IMAGE

## SHARPENED GRID IMAGE



**Why does Point Spreading Function (PSF) provide edge sharpening?**

Humans and other animals use vision to identify nearby objects. This is done by distinguishing one region in the image from another, based on differences in brightness and color. In other words, the first step in recognizing an object is to identify its edges, the discontinuity that separates an object from its background. An edge enhancement filter is used for this purpose.

The kernel undergoes shift and subtract operation. A shifted version of the image (corresponding to the –k/8) is subtracted from the original image (corresponding to the k+1). This processing distinguishes some objects that are closer or farther away than the background, making a 3D effect. The edge detection operation removes all the contrast leaving only the edge information.

Every edge in the original image is transformed into narrow dark and light bands that run parallel to the original edge. Thresholding this image can isolate either the dark or light band, providing a simple algorithm for detecting the edges in an image.

## SHARPEN.C VS SHARPEN_GRID.C

In Sharpen.c Code, the whole image is 300x400 as compared to the sharpen_grid.c which is 10 times that of the image resolution used in sharpen.ci.e 3000x4000.

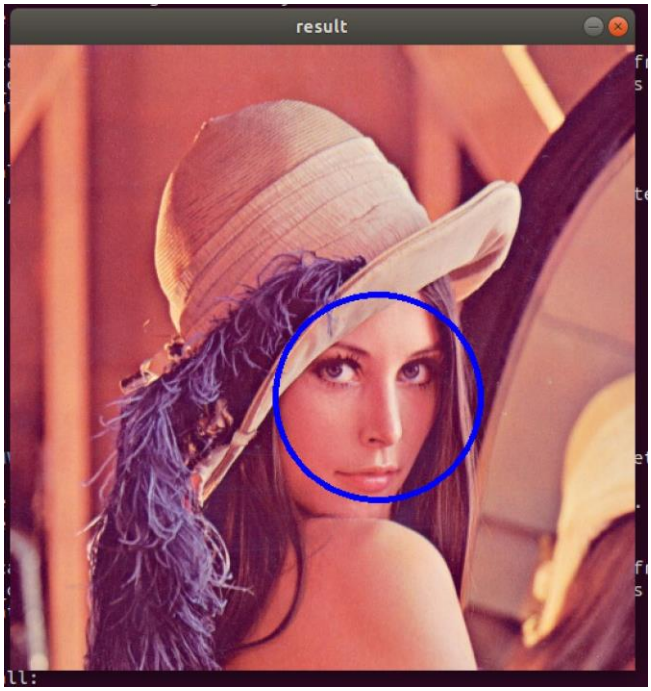
In sharpen.c the image to be transformed is read and stored in a one dimensional array. The RGB components are stored in different arrays. These values are then transformed by applying a Point spread function. The kernel is applied to each and every pixel obtaining the corresponding pixel value and thus, storingit in the respective converted RGB array. These values are then written to the output image to obtain the sharpened image.

Now since the sharpen_grid.c image had a much larger resolution as compared to the sharpen.c image, the whole image are divided into small grids which are transformed by individual threads thus utilizing all the cores on the CPU and making the process run faster. The same set of steps take place as in the sharpen.c explained above with the only difference that the image is divided in blocks of 500x500 and the same filter is applied by different threads. The threads are created with a callback function to sharpen that particular block of image. The RGB values are read on converted accordingly, the same way as it is done in sharpen.c , the only difference being here that a 2D array is used as opposed to 1D array used to store RGB values in sharpen.c. Finally, the values are then written to the output image to obtain the sharpened image. The same frame is sharpened 1000 times in order to have better edge definition.

# QUESTION 4

**ORIGINAL IMAGE**

## HAAR TRANSFORM IMAGE




## HAAR TRANSFORM OPERATION

Haar Cascade consists of a series of classifiers that help in detecting faces and other features in different faces. It is well known for detecting faces and body parts in an image but can be trained to identify almost any object. It is trained from a lot of positive and negative images. The way it works is it applies different features (filters) to separate the image into a positive or negative image. Initially features (filters) as many as 180000 were applied on 24x24 image which is a lot of computation. This was brought down to 6000 by using the concept of Adaboost training. Integral Images were also created in order to reduce the computations.

The algorithm has 4 stages: Haar Feature Selection, creating integral images, Adaboost training and cascading classifiers.

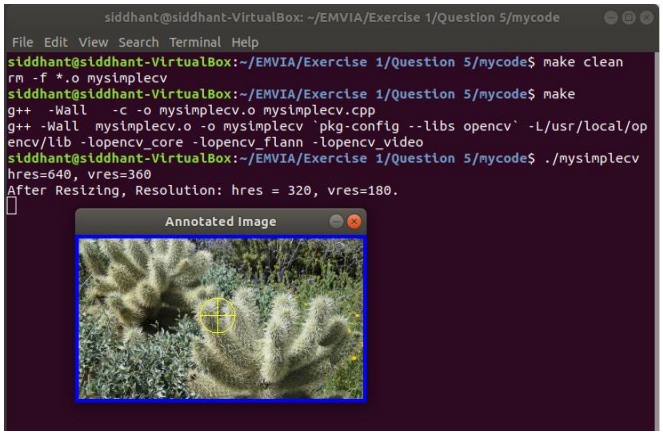
The Haar features sums up pixel values in adjacent rectangular regions and calculates difference between the sum of different regions. There can be various Haar features. This is made faster using Integral Images. Integral Images calculate the sum of all the matrices that can be formed inside an Image matrix. Thus having the sum of different regions beforehand would make it redundant to calculate it everytime at Haar feature processing. This would speed up the process to a great extent.

A fixed window is moved over the image with a fixed size. It employs haar features on the target image in a particular region. If the haar image gives a negative result, the window slides over to the next region and repeats the process. If the result is positive, it applies the subsequent features to correctly determine if it is a face or not. There are multiple features that need to be evaluated before it can be termed as a face. These large number of Haar features are necessary to describe an object with sufficient accuracy and are therefore organized into cascade classifiers to form a strong classifier.

# QUESTION 5

## ORIGINAL IMAGE

## ANNOTATED IMAGE




# REFERENCES


- https://opencv.org/
- https://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/sobel_derivatives/sobel_derivatives.html
- https://www.youtube.com/watch?v=uihBwtPIBxM
- https://www.youtube.com/watch?v=XuD4C8vJzEQ
- https://www.youtube.com/watch?v=uEJ71VlUmMQ
- http://www.willberger.org/cascade-haar-explained/
- http://www.dspguide.com/ch24.htm


# APPENDICES

The Folder consists of:

1. Code and Makefiles.
2. Screenshots and Images.