# ALGORITHMS AND OPERATING SYSTEMS PROJECT

- TEAM 8

## Project Details

The main objective of this project is to build a client-server application in which the client sends a video clip for face identification and the server runs the model to identify the faces in the video and returns the video with a bounding box on the face along with the name of the person.

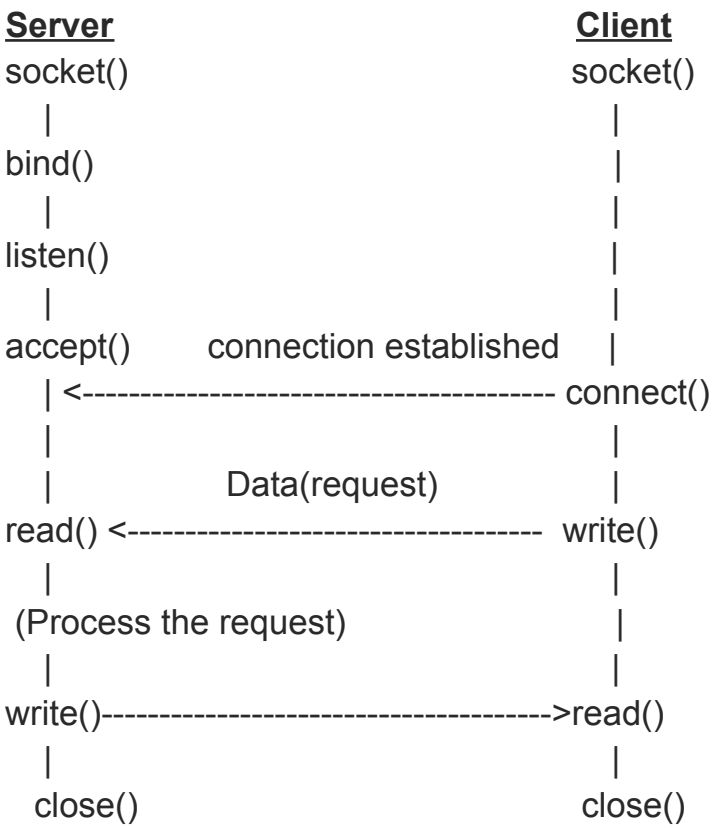
The image identification is done using viola-jones and skin segmentation algorithms together.

## Client-Server description

The server and the clients are implemented using Sockets in python. Sockets enable programs to send and receive data, bi-directionally, at any given moment.

The server is initialized with the address family - **Address Format Internet (**AF_INET**)** and its type format is **Socket Stream** which enables “sequenced, reliable, two-way, connection-based byte streams” over TCP. After creating the socket, we bind it with its port and address (5000 and localhost here respectively). The maximum number of connections that the socket can listen to at a time is set to 5.

The clients are initialized with the same address family and type as the server. A particular client opens up a socket connection with the server, but **only if the server program is currently running**.




**This is a general client-server model in TCP protocol**

The server supports the following commands:-

- Exit - To quit the server
- Listen - The server listens to the incoming message from a client. The server has to first accept the connection request from the client. For each client, a new thread is created. This way there are many threads of the same server serving the requests of different clients at the same time. Multitasking is achieved over here. The processing of requests of each client happens concurrently.
- trainVideo - The server receives the address of the video file along with the person's name (who is in the video) and is uses them for training
- trainWebc - The server takes the user's face and name as training data through a webcam.

The clients support the following commands:-

- Exit - To quit a particular client
- Video - To get the address of the video from the user which is supposed to be sent to the server for recognizing the person in the video. The server will return back the name of the identified person.
- Webcam - To get video of the user through the webcam. This video is sent to the server for recognizing the person in the video. The server will return back the name of the identified person.

In addition to the above points, the clients and server also report invalid commands.

## Description of the threading module

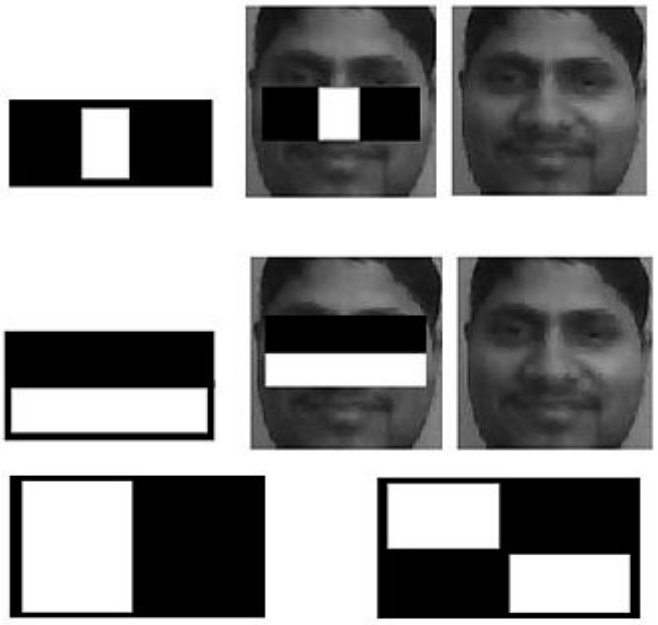
We have used the higher-level *threading* module of Python to implement threads. The higher-level model is built on the lower-level *thread* module. We create a new thread class in the *listen()* function of server whenever a new client-socket attempts to connect. The thread class has two important methods - *run()* and *start()*. When *run()* method is called, the current thread is executed(not the newly created one). When *start()* is used, the newly created thread is sent to *run()* method and it is executed.
The new thread class has objects - target and args. The target object is the callable object to be invoked by the *run()* method and *args* is the argument tuple for the target invocation. The target function for the thread is *clienthandler* which takes the client-socket as argument and returns the predicted name of the person in the image/video sent by the client. The is_alive() method checks if the thread is killed or not.

## Face Identification Algo

## Viola-jones

Viola-jones detect faces in the image and the skin segmentation algorithm identifies the faces detected by viola-jones. Viola-jones uses an ensemble learning technique that uses a mixture of weak classifiers which are basically haar features and combines them to form a strong classifier. It is trained using a technique called ada boost learning and it computes the integral image to make finding the sum of a given area quicker. Finally, it cascades the classifiers with a lesser number of features used at the upper levels so that windows without faces can be filtered out quickly and it uses more features in the lower levels so that the face finding accuracy increases. Each cascade layer can be used to detect different parts of faces like the left eye, right eye, etc.

## Haar Features



All human faces share some similar properties. These regularities may be matched using Haar Features.

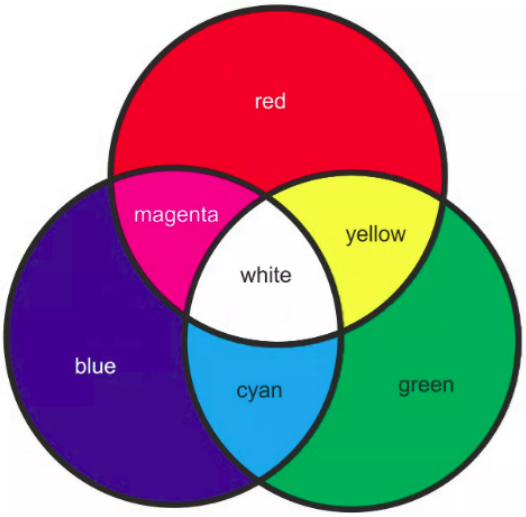
## Skin Segmentation


In case the cascaded classifiers fail in detecting a face in the image, then skin segmentation method is used in order to identify the face.

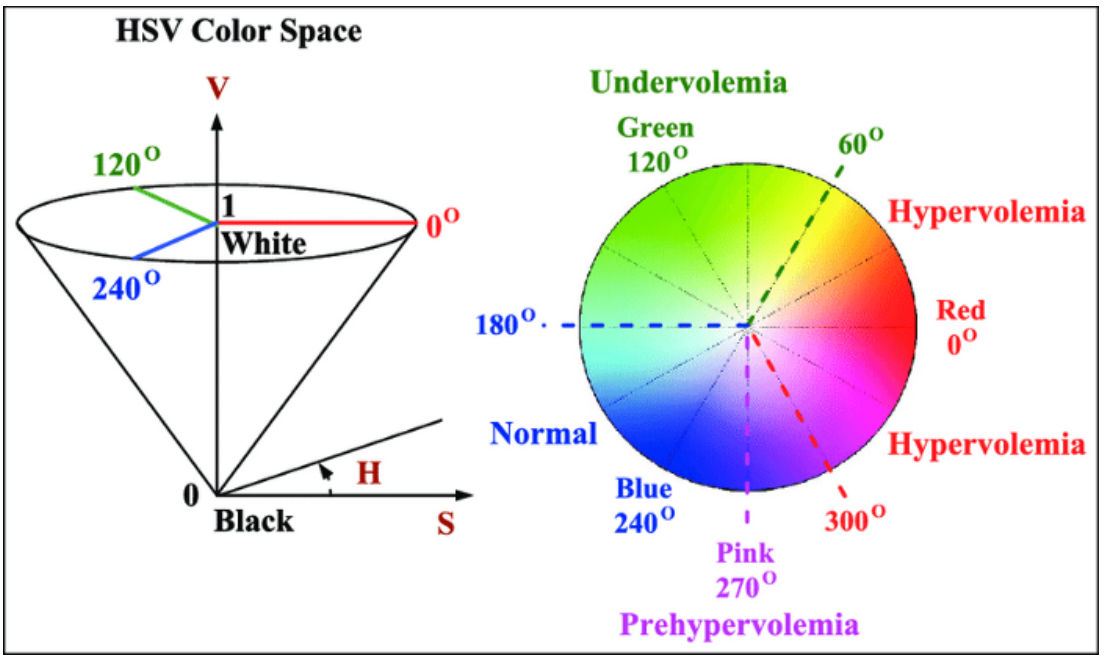
Human skin color detection, for the most part, is used for human face detection. It is one of the best approaches for finding and extracting a person’s face in a given picture. Often it is understood that, for individuals of different races, the fragments' skin tones change to some degree. To feature the color of human skin, different color spaces are accessible.

The three primary spaces for recognizing a skin are RGB (Red, Green, Blue), YCbCr (Luminance, Chrominance), and HSV (Hue, Saturation, Value) color models.

## Color Spaces



RGB color model




We compute masks for each color space and apply them to the image to isolate the skin texture.

### 1) RGB Space



\

2) **YCbCr Space**

**3) Hue Mask**

If we combine all of the previous masks, the algorithm will generate the RGB-YCbCr-HSV mask, and if we apply this mask on a given image it will generate the required output

**LBPH algorithm**

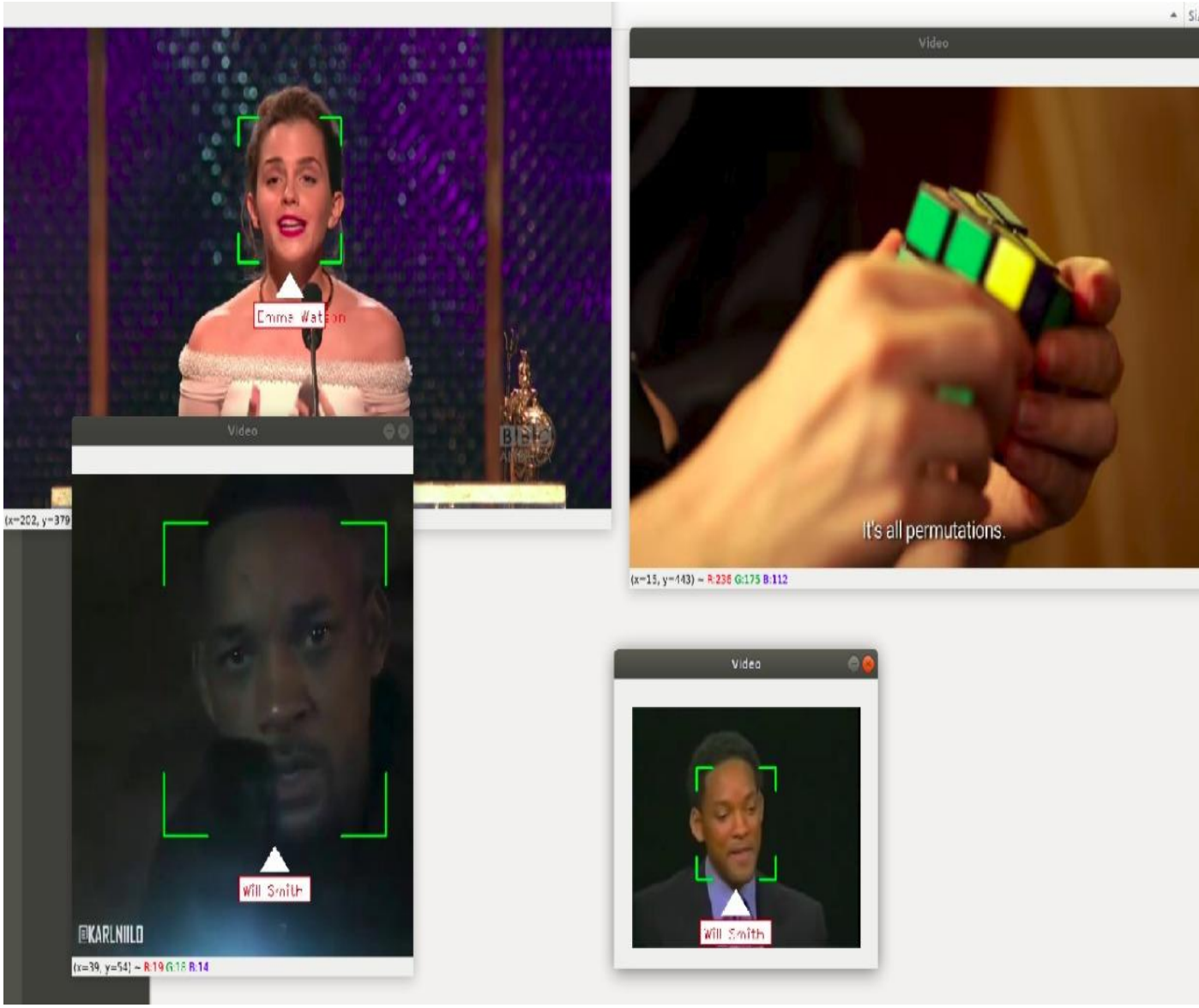
LBPH algorithm is used to identify faces detected by viola-jones in the previous step. It is initially trained on the sample faces of persons to be identified and can be tested on other photos of the same person. The algorithm has 4 parameters named radius, neighbors, GridX and GridY. Initially, a grayscale image is taken and the neighboring pixels within a given radius(radius and neighbors are parameters) are considered and are marked binarily using the center pixel value as the threshold. All these binary values are concatenated and the pixel value is replaced by this number. Finally, a histogram is prepared for this image with the gridX and gridY values to divide the image into smaller grids. By now the model is trained. Now to test the model using an image, we again compute the histogram of the image using the steps used above and the similarity between histograms is computed using the formula below.

$$D = \sqrt{\sum_{i=1}^{n}(hist1_i - hist2_i)^2}$$

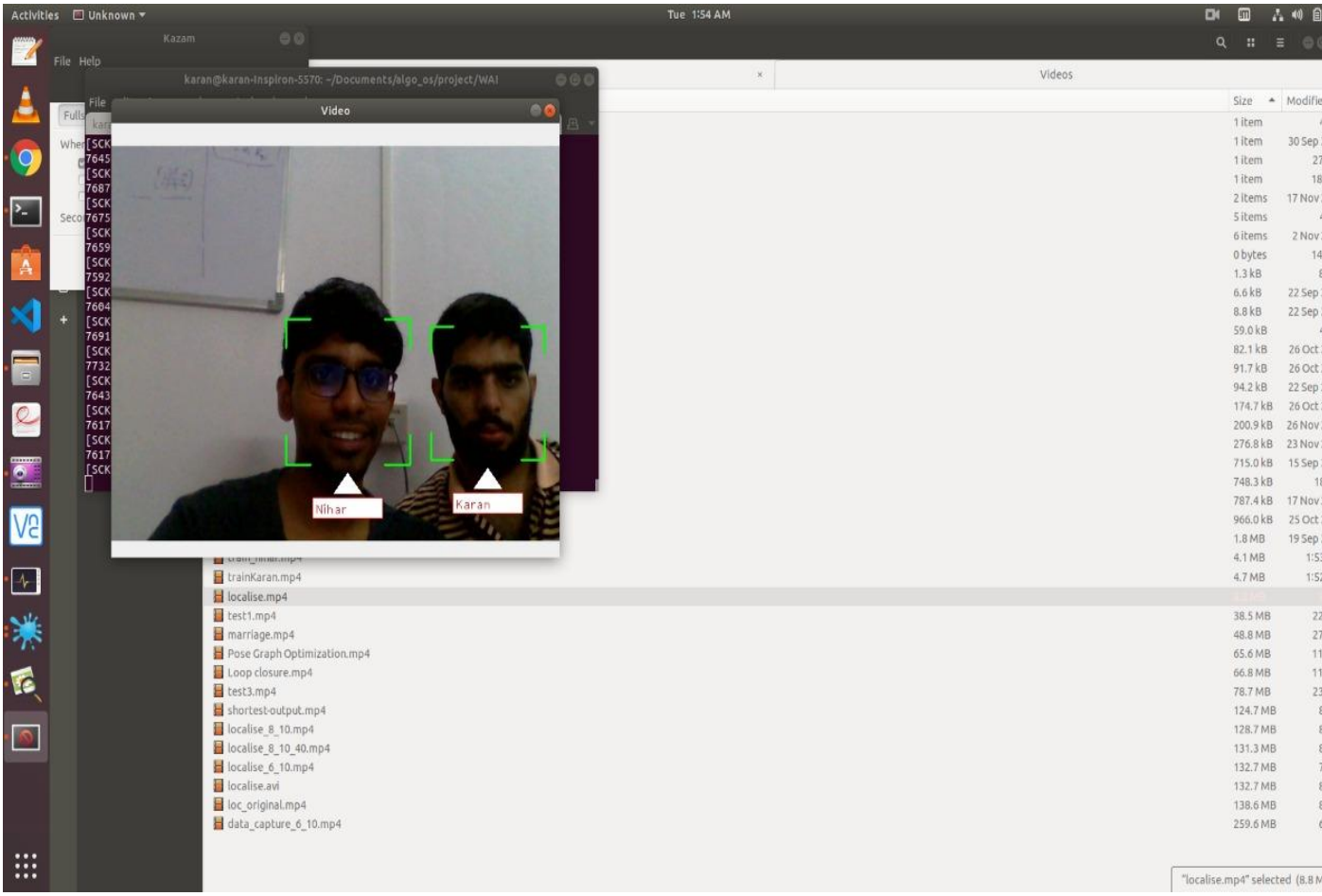
The similarity is computed with the histograms of all trained images and if the similarity is more than a certain threshold with some image, then it can be concluded that a match is found with the target image.

## Results

1. Multiple Clients handled at the same time



2. Result from Webcam




3. Videos -

1. https://youtu.be/Uy-cnb47BpA
2. https://youtu.be/R7PfAs8LVPE