“VIDEO X-RAYING” - CHARACTER AND OBJECT RECOGNITION IN VIDEOS FOR ENHANCED VIEWING EXPERIENCE

A PROJECT REPORT

Submitted by

POORANI R. **312214106070**

PRADEEP G. **312214106071**

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

ELECTRONICS AND COMMUNICATION ENGINEERING

SSN COLLEGE OF ENGINEERING

ANNA UNIVERSITY: CHENNAI 600 025

APRIL 2018

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report titled "**VIDEO X-RAYING" - CHARACTER AND OBJECT RECOGNITION IN VIDEOS FOR ENHANCED VIEWING EXPERIENCE** is the *bonafide* work of **POORANI R. (312214106070)** and **PRADEEP G. (312214106071)**, who carried out the project work under my supervision.

Dr. S. Radha
Head of the Department
Professor,
Department of Electronics and Communication Engineering,
SSN College of Engineering,
Kalavakkam- 603110.

Dr. S. Joseph Gladwin
Supervisor
Associate Professor,
Department of Electronics and Communication Engineering,
SSN College of Engineering,
Kalavakkam- 603110.

Place:

Date:

Submitted for the examination held on _____

Internal Examiner External Examiner

ACKNOWLEDGEMENTS

This project could not have been completed without the selfless help of various professors and friends during the past few months. We express our deep sense of gratitude to our guide **S. Joseph Gladwin**, Associate Professor, ECE Department, for his valuable advice and suggestions as well as his continued guidance, patience and support that helped us to shape and refine our work. Our thanks to **S. Radha**, Professor and Head of the Department of Electronics and Communication Engineering, for her words of advice and encouragement. We also thank our project coordinator **Venkateswaran N.,** Professor, ECE Department, and the other panel members for their valuable suggestions during the project reviews.

We express our deep respect to the founder **Shiv Nadar**, Chairman, SSN Institutions and would like to thank him for being a source of inspiration. We also appreciate **S. Salivahanan**, Principal, for all the help he has rendered during this course of study.

We extend our sincere thanks to all the teaching and non-teaching staffs of our department who have contributed directly and indirectly during the course of this project. We are highly indebted to our parents, for encouraging and supporting us all through. Thanks as well to fellow students for their timely help with various problems during the course of this project.

Poorani R.

Pradeep G.

ABSTRACT

Online video streaming has undergone exponential growth over the past few years. Numerous video streaming platforms like Netflix, Amazon Prime Video and Hotstar seamlessly stream TV shows, movie premieres, live sports and other events from around the world, thus, captivating millions of viewers. In spite of the relative advantages of such online platforms over conventional off-line platforms, viewing experience is still not satisfyingly great. The major reason is the unavoidable series of commercial advertisements during the course of the video, which also destroys the ultimate purpose of effective marketing and turns into a miscarriage for the advertiser itself. Another unfavorable instance frequently encountered in online platforms is the effort that viewers have to take to know about the characters, music or trivia of the particular video as it requires additional browsing, which makes it all the more inconvenient.

In this project, we enhance the viewing experience by solving the above two issues. To eliminate boring intermissions, products to be advertised are detected during the course of the video itself using SIFT features and an easy option is provided to buy them online with the help of web scrapping and an interactive Graphical User Interface (GUI). To eliminate the need for additional browsing to get more information about the characters, face recognition is employed and information is displayed about the character by clicking on the character's face during video playback.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

"Any sufficiently advanced technology is indistinguishable from magic." says Arthur C Clarke. Just like magic, technology has now become a tool to seek comfort and convenience, rather being used solely to meet basic necessities. With the advent of technology, humans have tried to transform the good into better and the better into best. In harmony to this, one concern that requires attention is the difficulties people face while watching a video online in spite of the relative advantages like instant playback, storage facility, technology-related attributes and other key perceived characteristics over conventional off-line platforms.

To analyze different perspectives of this, we conducted a survey among 50 adults, who frequently watch online videos. During the survey, the respondents were asked to list the inconveniences while watching videos online and indicate the level of agreement for the statements posed. From the results of this survey depicted in Figure 1, it is clear that two of the major issues are:

1. Unavoidable waiting for the intermissions for advertisements, which become a source of nuisance for the viewers, to end, and

2. Impossibility of gathering information about the characters, music and trivia of the particular video at that instant, which is possible only through additional browsing later.

The latter issue was sort of addressed by Amazon X-Ray feature implemented in Kindle Amazon to access the general reference information such as dictionary, encyclopedia and information about characters featured in stories without using internet. This feature was further upgraded in Amazon Prime Videos to display the IMDb profile of characters, and information about music and trivia of the video instantaneously while watching it without disturbing viewer's experience. Our idea is to employ face recognition and display information about the character by clicking on the character's face during video playback itself. We extend this to a further step by providing an advertising platform during the course of video itself, instead of having an intermission to advertise products. This is done by employing object detection algorithms on the video feed and providing an easy option to buy them online with the help of web scrapping and an interactive Graphical User Interface (GUI).

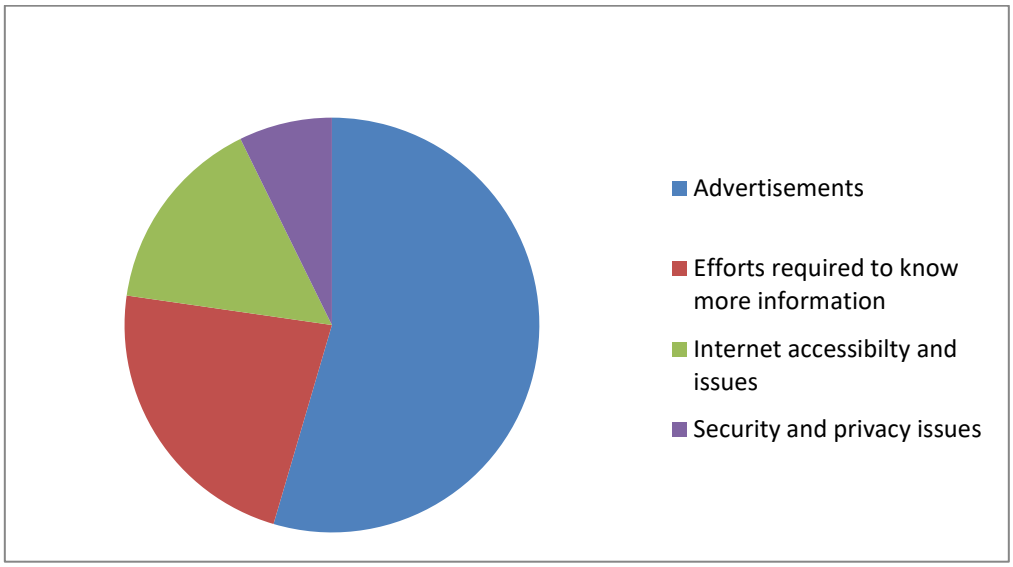

Fig. 1: Pie chart depicting the different problems faced by online viewers

1.2 SIGNIFICANCE OF THE PROBLEM STATEMENT

Advancements in science and technology have led to the implementation of online video streaming surpassing offline video streaming platforms. Online video streaming facilitates streaming in any compatible electronic device at any time with an access to internet connection, because of which there is a shift towards such online platforms. Aware of this drastic shift, industrialists now give more importance to advertise their products online in order to attract viewers, who watch online videos, and form a huge audience of consumers for themselves. This strategy, though seems wise, does not imply so. Moreover, an industry report shows that consumption of advertising supported online shows will be 10 times more than that of paid online videos (Hopewell, 2009) as people prefer not to spend money for such luxuries. But, watching advertisements every now and then while streaming is of great nuisance and it greatly disturbs the viewer experience. Thus, the ultimate purpose of such advertisements is not effective and does the opposite of what it intends to. This points out the fact that advertising a product has become so crucial in today's world, but, at the same time, it has become difficult for the advertiser to reach out effectively to the audience and for the public to make use of such time-consuming advertisements.

The diffusion of innovations theory is widely used to explain the adoption of a new media technology. Diffusion of innovations theory suggests that consumers experience persuasion due to perceived characteristics of an innovation; these perceived characteristics are important factors that affect the adoption of the innovation. The flexibility of this new technology has opened up online platforms to a wider range of audience. But, they do not seem

satisfied with that flexibility has internet has no limitation. They expect to know about the characters of a particular series or movie while watching it, with even more curiosity in knowing about the real lives of their celebrities. Unfortunately, this cannot be done instantly. They can only additionally browse either later or after pausing the video. But, what if one can get such information just by mere pausing of the video without any further browsing? This is exactly the objective of our problem statement.

1.3 ORGANIZATION OF THE REPORT

Chapter 2 describes the previous literature on the topic and provides an overview of the different algorithms used previously. Chapter 3 emphasizes the concepts behind face detection and presents the fundamentals of face recognition using different methods, highlighting the pros and cons of each of them. Chapter 4 introduces the different types of conventional object detection techniques and details the Scale Invariant Feature Transform algorithm we have incorporated in our project. Chapter 5 presents the implementation details and the performance of our system. Chapter 6 gives a brief conclusion of our project.

CHAPTER 2

LITERATURE SURVEY

2.1 EARLY APPROACHES FOR FACE RECOGNITION

For the past 30 years, many researchers have proposed different techniques for face recognition, motivated by the increased number of real world applications. There are several problems that make face recognition in real time a hectic task [1]. To make face recognition robust, the input dataset is taken under different conditions with changes in the following parameters such as pose, illumination, expression, motion, facial hair, glasses, and background [2][3].

To recognize a face in a given image, neural network based approaches were used. The main idea was to use machine learning to find the relevant characteristics in the image. The learned characteristics, in the form of discriminant functions (i.e. non-linear decision surfaces), are subsequently used for face recognition. Conventionally, face images are projected to a low-dimensional feature space and a non-linear decision surface is formed using multilayer neural networks for classifications and recognition [10]. The advantage of using neural networks for face recognition is that they can be trained to capture more knowledge about the variation of face patterns, and thereby achieve good generalization [13].

The main drawback of this technique is that the networks have to be extensively tuned to get exceptional performance. Among the neural networks approaches for face recognition, Multilayer Perceptron (MLP) with Back

Propagation (BP) algorithm has been mostly used [14]. However, the convergence of the MLP networks is slow and the global minima of the error space may not be always achieved [11]. On the other hand, the Radial Basis Function (RBF) neural networks have fast learning ability [15] and best approximation property [16]. So, in recent times, many researches have used RBF networks for face recognition. However, their success rates are not promising as the error rates vary from 5% to 9% under variation of pose, orientation, scale and light [13]. This may be due to the fact that the selection of the centers of the hidden layer neurons might not have been done by capturing the knowledge about the distribution of training patterns and variations of face pose, orientation and lighting [19].

Later, in 1991, Turk and Pentland used Principal Component Analysis (PCA) projections as the feature vectors to solve the problem of face recognition, using the Euclidean distance as similarity function [20]. This system, later called Eigen Faces, was the first Eigen space-based face recognition approach and, from then on, many Eigen space-based systems have been proposed using different projection methods and similarity functions.

Eigen Face recognizer looks at all the training faces of all the input faces at once and finds the combined principal component from all of them. By capturing this combined principal component, one is not focusing on the features that discriminate one person from the other, but on the features that represent all the people in the training data as a whole [26].

Face recognition using Fisher Faces is an improved version of Eigen Faces [25]. Fisher Faces algorithm, instead of extracting features that represent all the faces of all the people, it extracts useful features that discriminate one person from the other. By this way, features of one person

do not dominate over the others and we end up having features that discriminate one person from the other [27]. It is observed that even in Fisher Faces algorithm, if multiple people have faces with sharp changes due to external sources like light, then these variations will dominate over other critical features and will affect recognition accuracy. In [], from the evaluation, it was observed that Eigen Face was better than Fisher face with constant lighting conditions by comparing the Region of Convergence (ROC) curve.

Linear Binary Patterns Histogram (LBPH) recognizer overcame the drawbacks of both Eigen Face and Fisher Face recognizers, as it exploited the idea to not look at the image as a whole; instead, to find the local features of an image by comparing each pixel with its neighboring pixels.

2.1 EARLY APPROACHES IN OBJECT DETECTION

Object detection and tracking is one of the critical areas of research due to constant change in motion between object of interest and frame, variation in scene size, occlusions, appearance variations, ego-motion and illumination changes. Object detection basically means to check the existence of that object in the frame. Then, classifying the detected object into various categories such as humans, vehicles or other moving objects is object recognition [2].

The techniques stated in [3] for object detection ranges from very basic algorithms to the state of the art published techniques, categorized based on factors like speed, memory requirements and accuracy. The different techniques were frame difference technique, real time background subtraction and shadow detection technique, and adaptive background

mixture model for real time tracking technique. They were also able to deal with real time challenges like snow, rain, moving branches, objects overlapping, light intensity and slow moving objects.

There are three basic phases in video examination: detection of interesting objects in video scene, tracking of such objects from frame to frame, and analysis of object tracks to recognize their activities. Detecting humans from a video is a challenging problem owing to the motion of the subjects as discussed above. In [6], they developed a detector for moving people in videos with possibly moving cameras and backgrounds, testing several different coding schemes for moving objects and showed that orientated histograms of differential optical flow only gives the maximum performance [14].

Moving object detection technique as shown in fig 2, frame difference technique and the approximate median method were used to detect objects in fast moving videos [7]. Moving object detection and object tracking by using the modified frame difference method is now being employed in various places.

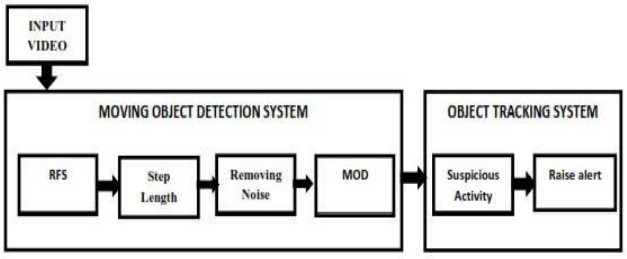

Fig. 2: Block diagram of object detection & tracking

Cascade-of-rejecters approach along with Histograms of Oriented Gradients (HOG) features were also used to achieve a fast and accurate

human detection system [5]. The features used are histograms of oriented gradients of variable-size blocks that capture salient features of humans automatically. Feature selection is then employed to identify the appropriate set of blocks, from a large set of possible blocks. It uses the integral image representation and a rejection cascade, which significantly speed up the computation. For an image, the system can process 5 to 30 frames per second depending on the density in which it scans the image, while maintaining an accuracy level similar to existing methods.

However, all the above discussed methods have some limitations, in certain circumstances which are listed as follows [8].

Lightning: Low light adds darkness to an image, while excess light adds shadowing effect to the image.

Positioning: Uniform position is required throughout or else it is unable to detect object even if it is present in the image.

Rotation: Image can be rotated in any direction. In this case, if shape matching method is used, then some shapes are unable to be identified unless it is symmetrical.

Occlusion: An object behind another object is sometimes not completely visible and so it cannot be detected.

CHAPTER 3

FACE DETECTION AND RECOGNITION

3.1 FACE DETECTION

3.1.1 INTRODUCTION

For a face to be recognized in a video, it needs to be detected first. For something as complicated as a face, there isn't one simple test that will tell if there is a face or not. The algorithm has to break the task of identifying the face into thousands of smaller, bite-sized tasks, each of which is easy to solve. These tasks are named as classifiers. For detecting a face in a given image, the algorithm must scan the image from the top left corner and move across the image assuming it to consist of small blocks of data and constantly checking if each block is a face or not. Since there are many blocks, there will be millions of computations to be performed, which will grind the computer to a halt.

To get around this, we used the concept of cascades. Like a series of waterfalls, the cascade breaks the problem of detecting faces into multiple stages. For each block, it does a very rough and quick test. If that passes, it does a slightly more detailed test, and so on. The algorithm will have around 30-50 of these stages or cascades, and it will only detect a face if all stages pass. The advantage is that the majority of the pictures will return negative during the first few stages itself, which means the algorithm would not waste

time testing all the features. So, instead of taking hours, face detection can now be done in real time.

3.1.2 CASCADES IN PRACTICE

Though the theory may sound complicated, in practice it is quite easy. Object Detection using Haar feature-based cascade classifiers is an effective object detection method proposed by Paul Viola and Michael Jones in their paper, “Rapid Object Detection using Boosted Cascade of Simple Features” in 2001. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images.

3.1.3 WORKING PRINCIPLE

Initially, a huge set of lot of positive images and negative images is fed to train the classifier. Positive images are those images which contain a face in it. On the other hand, any image without a face can be taken as a negative image. The next step is to extract features from the pre-fed images. For this, Haar features, shown in the below image, were used. They are just like the convolutional kernels. Each feature is a single value obtained by subtracting sum of pixels under the white rectangle from sum of pixels under the black rectangle as shown in Figure 3.

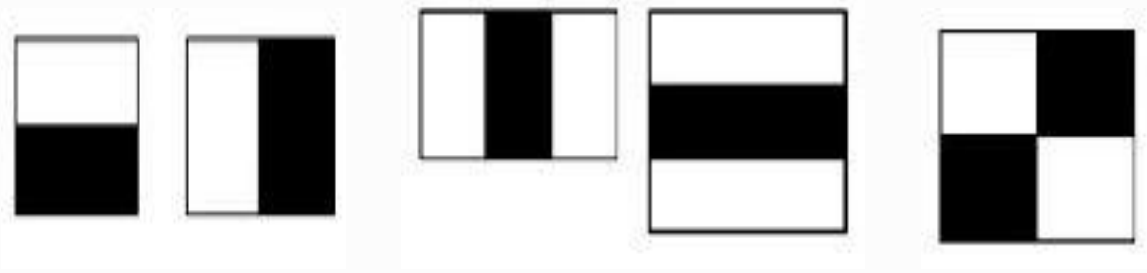

Fig. 3: Extracting features using rectangles

Now, all possible sizes and locations of each kernel is used to calculate plenty of features. For example, even a 24x24 window results over 160000 features. For computing each feature, we need to find the sum of pixels under white and black rectangles, which amplifies the task. To solve this, they introduced the concept of integral images. It simplifies the calculation of the sum of, even, a large number of pixels. This way, the speed of the system is greatly improved.

But, among the features calculated, most of them are irrelevant. For example, consider Figure 4. The top row shows two good features. The first feature selected seems to focus on the property that the region of the eyes is often darker than the region of the nose and cheeks. The second feature selected relies on the property that the eyes are darker than the bridge of the nose. But, applying the same windows on cheeks or any other place will be irrelevant. So, to make it easier and find out the right feature, Ada Boost algorithm is used.

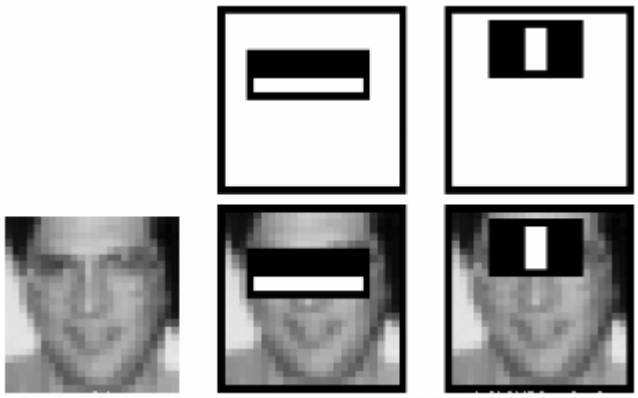

Fig. 4: Extraction of right features from a face

For this, each and every feature is applied on all the training images. For each feature, it finds the best threshold which will classify the faces to positive and negative. Then, the system selects the features with minimum error rate, which means they are the features that best classifies the face and

non-face images. Each image is given an equal weight in the beginning. After each classification, weights of misclassified images are increased. Then, the process is repeated. New error rates and new weights are calculated. The process is continued until required accuracy or error rate is achieved or required number of features is found.

Final classifier is a weighted sum of these weak classifiers. They are called weak because they cannot be used to classify the image on their own, but together with others will form a strong classifier. But, it is still inefficient and time consuming. In an image, most of the image region is non-face region. So, it is a better idea to have a simple method to check if a window is not a face region. If it is not, it is discarded in the first classifier itself. Instead, focus is given on the region with the probability of having a face. For this, the concept of Cascade of Classifiers was introduced. This way, the time required to detect a face in a given image is greatly reduced and the face can be detected in a fraction of a second as shown in Figure 5.

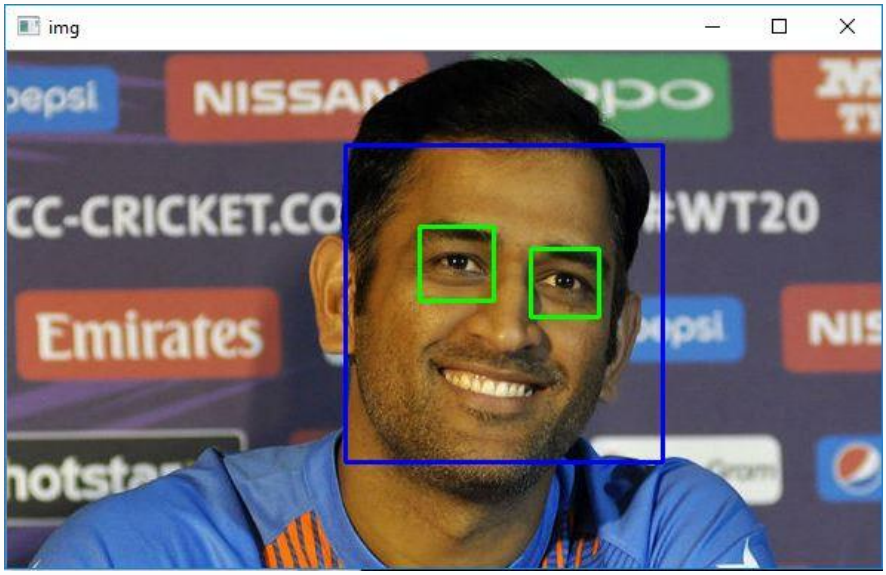

Fig. 5: Face detection in an image

3.2 FACE RECOGNITION

3.2.1 INTRODUCTION

One's prominent facial features are used to recognize to whom that face belongs to. The facial features vary from person to person and that forms the basis of distinction between several people. Take a real life example, when we meet someone for the first time, we do not recognize them. But by looking into their face, eyes, nose, mouth, colour and overall look daily, we are able to do it. This is how our mind is learning or training for recognizing the face of that person by mere data gathering. Once the mind is trained to recognize the face, then the next time when we will see the same face, we will be able to recognize immediately. Now, this is the case with respect to humans. Computers are not of much difference; more or less, they follow same ideology to recognize faces. Training a computer to recognize faces follows three main steps:

- Training Data Gathering: Gathering of face data (face images in this case) of people to be recognized
- Training of Recognizer: Feeding that face data (and respective names of each face) to the face recognizer so that it can learn.
- Recognition: Feeding new faces of the people and seeing if the face recognizer recognizes them accurately.

For making things much easier, we used a computer vision library "Open CV with python bindings" and programmed using Python. Open CV (Open Source Computer Vision) is a popular computer vision library started by Intel in 1999. The cross-platform library sets its focus on real-time image

processing and includes patent-free implementations of the latest computer vision algorithms.

3.2.2 OPENCV FACE RECOGNIZERS

Open CV has three built-in face recognizers namely,

1. Eigen Faces Face Recognizer
2. Fisher Faces Face Recognizer
3. Local Binary Patterns Histograms (LBPH) Face Recognizer

3.2.3 EIGEN FACES FACE RECOGNIZER

3.2.3.1 THEORY

This algorithm considers the fact that not all parts of a face are equally important and equally useful. Distinct features like eyes, nose, cheeks and forehead vary with respect to each other. So, focusing on the areas of maximum change (mathematically speaking, this change is variance) of the face is paramount. For example, from eyes to nose there is a significant change and same is the case from nose to mouth. Looking at multiple faces, we compare them by looking at these parts of the faces because these parts are the most useful and important components of a face. Important because they catch the maximum change among faces, change that helps one differentiate one face from the other. This is exactly how Eigen Faces face recognizer works.

Eigen Faces face recognizer looks at all the training images of all the people as a whole and tries to extract the components, which are important and useful, and discards the rest of the components. This way it not only extracts the important components from the training data, but also saves memory by discarding the less important components. The important components extracted are called principal components. Figure 6 shows the principal components extracted from a set of faces.

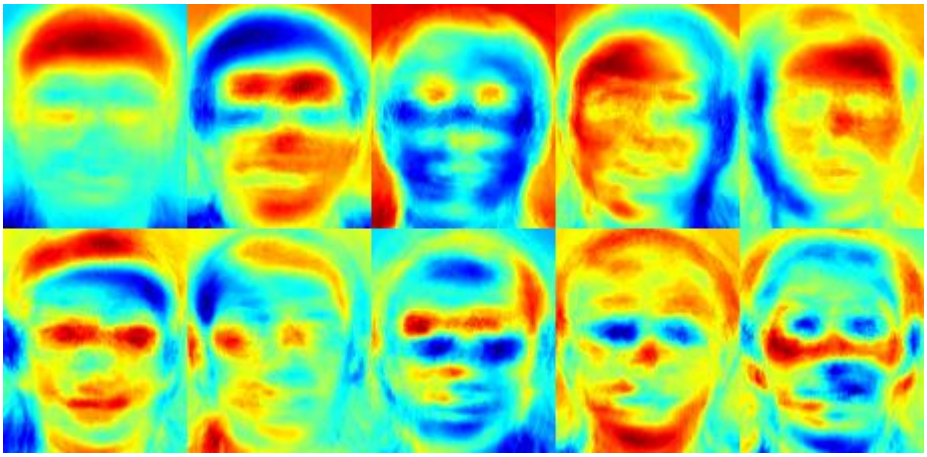

Fig. 6: Principal components of faces

Principal components actually represent faces and these faces are called Eigen faces and hence, the name of the algorithm. So this is how Eigen Faces face recognizer trains itself by extracting principal components. It also keeps a record of which principal component belongs to which person. But, unfortunately, Eigen faces algorithm also considers illumination as an important component.

Later during recognition, when a new image is fed to the algorithm, it repeats the same process on that image as well. It extracts the principal component from that new image and compares that component with the list of

components it stored during training. It, then, finds the component with the best match and returns the person label associated with that best match component.

3.2.3.2 MATHEMATICAL EXPLANATION OF THE ALGORITHM

Let $\boldsymbol{X} = \boldsymbol{X_1}, \boldsymbol{X_2}, \ldots, \boldsymbol{X_n}$ be a random vector with observations $\boldsymbol{x_i} \in \boldsymbol{R^d}$

1. Compute the mean of all the components

$$\mu = \frac{i}{n} \sum_{i=1}^{i} x_i$$

2. Compute the Covariance Matrix S

$$S = \frac{1}{n} \sum_{i=1}^{n} (x_i - \mu)(x_i - \mu)$$

3. Compute the eigenvalues γ_i and eigenvectors v_i of S

$$Sv_i = \gamma_i v_i, i = 1,2,\ldots,n$$

4. Order the eigenvectors descending by their eigenvalue. The k principal components are the eigenvectors corresponding to the k largest eigenvalues.

The k principal components of the observed vector x are then given by:

$$y = W^T(x - \mu)$$

Where, $W = (v_1, v_2, \ldots, v_k)$

The reconstruction from the PCA basis is given by:

$$x = W_y + \mu$$

Where, $W = (v_1, v_2, \ldots, v_k)$

The Eigen faces method then performs face recognition by:

- Projecting all training samples into the PCA subspace.
- Projecting the query image into the PCA subspace.
- Finding the nearest neighbour between the projected training images and the projected query image.

There is yet another problem left to be solved. Imagine we are given 400 images sized $100 * 100$ pixels each. The Principal Component Analysis solves the covariance matrix $S = XX^T$, where $size(X) = 100000\ X\ 40$ in this example. We would end up with a $10000\ X 10000$ matrix, roughly $0.8\ GB$ in this way. Solving this is not feasible, so from linear algebra, it is clear that a $M\ X\ N$ matrix with $M > N$ can only have $N - 1$ non-zero eigenvalues. So it's possible to take the eigenvalue decomposition $S = X^T X$ of size $N\ X\ N$ instead:

$$X^T X V_i = \gamma_i$$

and get the original eigenvectors of $S = X^T X$ with a left multiplication of the data matrix:

$$X X^T X V_i = \gamma_i$$

The resulting Eigen vectors are orthogonal; Normalizing them to unit length gives orthonormal Eigen vectors.

3.2.4 FISHER FACES FACE RECOGNIZER

3.2.4.1 THEORY

This algorithm is an improved version of Eigen Faces face recognizer. Eigenfaces face recognizer looks at all the training faces of all the persons at once and finds principal components from all of them combined. By capturing principal components from all of them combined, the features that discriminate one person from the other are not focussed, but the features that represent all the persons in the training data as a whole.

This approach has drawbacks, for example, images with sharp changes (like light changes which is not a useful feature at all) may dominate the rest of the images and features that are from external source like light and are not useful for discrimination at all may be the result. In the end, the principal components will represent light changes and not the actual face features.

Fisher Faces algorithm, instead of extracting useful features that represent all the faces of all the persons, it extracts useful features that discriminate one person from the others. By this way, features of one person do not dominate over the others and the features that discriminate one person from the other are obtained. Figure 7 is an image of features extracted using Fisher Face algorithm.

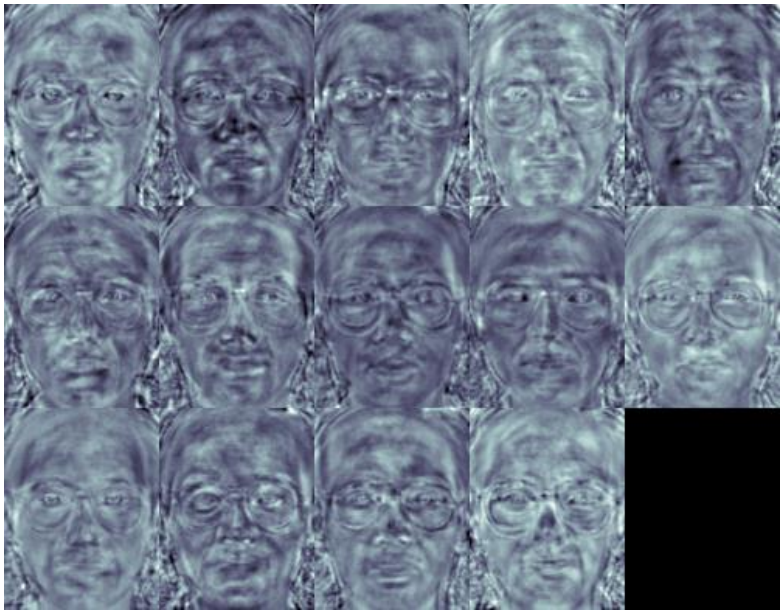

Fig. 7: Features extracted using Fisher Face algorithm

Features extracted actually represent faces and these faces are called fisher faces and hence, the name of the algorithm is Fisher faces. Even in Fisher Faces algorithm, if multiple people have images with sharp changes due to external sources like light, then they will dominate over other features and affect recognition accuracy.

3.2.4.2 MATHEMATICAL EXPLANATION OF THE ALGORITHM

Let X be a random vector with samples drawn from c classes:

$$X = \{X_1, X_2, \ldots . X_n\}$$

$$X_i = \{X_1, X_2, \ldots . X_n\}$$

The scatter matrices S_B and $S\_\{W\}$ are calculated as:

$$S_B = \sum_{i=1}^{C} N_i \, (\mu_i - \mu)$$

$$S_W = \sum_{i=1}^{C} N_i (x_i - x)$$

, where μ is the total mean:

$$\mu = \frac{1}{N} \sum_{i=1}^{N} x_i$$

And μ_i is the mean of class: $i \in \{i, \ldots, c\}$:

$$\mu_i = \frac{i}{X} \sum_{x_j} x$$

Fisher's classic algorithm now looks for a projection W, that maximizes the class separability criterion:

$$W_{opt} = \arg max_w \frac{SW^T S_B}{W^T S W_W}$$

A solution for this optimization problem is given by solving the General Eigenvalue Problem:

$$S_B V_i = \gamma_i S_w V_i$$
$$S_w^{-1} S_B V_i = \gamma_I V_i$$

There's one problem left to solve: The rank of S_w is at most $(N - C)$, with N samples and c classes. In pattern recognition problems the number of samples N is almost always smaller than the dimension of the input data (the number of pixels), so the scatter matrix S_w becomes singular. This was solved by performing a Principal Component Analysis on the data and projecting the samples into the $(N - C)$, -dimensional space. A Linear Discriminant Analysis was then performed on the reduced data, because S_w isn't singular anymore. The optimization problem can then be rewritten as,

$$W_{pca} = \arg max_w \, (W^T S_B W)$$

$$W_{fld} = argmax_w \, \frac{W^T W_{pca}^T S^B W_{PCA}}{W^T W_{pca}^T S_W W_{pca}}$$

The transformation matrix W, that projects a sample into the $(c-1)$-dimensional space is then given by:

$$W = W_{fld}^T W_{pca}^T$$

3.2.5 LOCAL BINARY PATTERNS HISTOGRAMS (LBPH) FACE RECOGNIZER

3.2.5.1 THEORY

Eigen Faces and Fisher Faces are both affected by light and in real life we cannot guarantee perfect light conditions. LBPH face recognizer is an improvement to overcome this drawback.

The idea is to not look at the image as a whole; instead, to find the local features of an image. LBPH algorithm tries to find the local structure of an image and it does that by comparing each pixel with its neighbouring pixels.

A 3x3 window is taken as shown in figure 8 and moved on the image, at each move (each local part of an image), the pixel at the centre is compared with its neighbouring pixels. The neighbours with intensity value less than or equal to centre pixel are denoted by 1 and others by 0. Then these 0/1 values fewer than 3x3 windows are read in a clockwise order and a binary pattern like 11100011 is formed. This pattern is local to some area of the image. Once this is done on the whole image and we will have a list of local binary patterns.

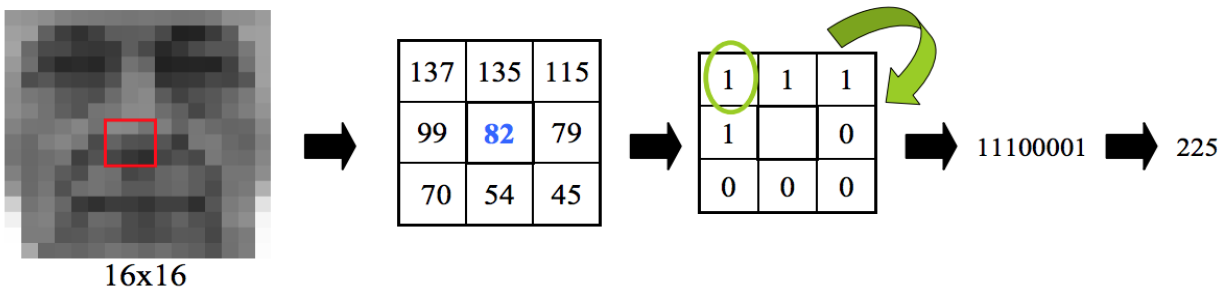

Fig. 8: LBPH algorithm

This algorithm has Local Binary Patterns in its name because a list of local binary patterns is generated. Later, each binary pattern is converted into a decimal number as shown in Figure 8 and then a histogram of all of those values is made. A sample of histogram generated is shown in Figure 9.

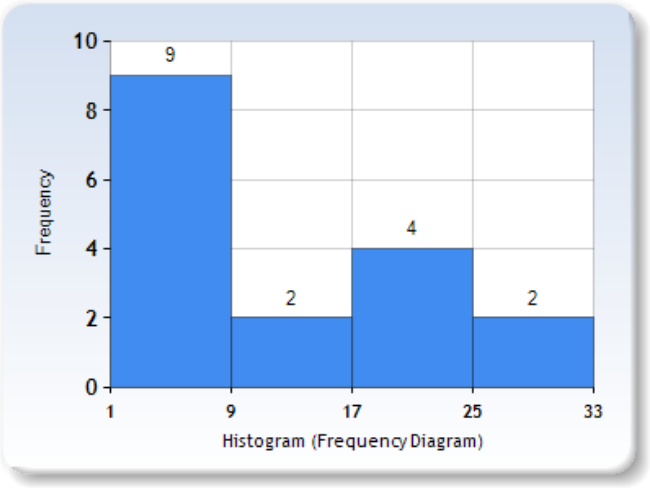

Fig. 9: Sample Histogram

So, in the end, one histogram for each face image in the training data set is created. The algorithm also remembers which histogram belongs to which image of which person.

Later, when a new image is fed for recognition, the histogram of that image is extracted and then compared with the existing histograms in the database. This is how a person's face is recognised using this algorithm. Figure 10 shows a list of faces and their respective local binary patterns images. LBP images are not affected by changes in light conditions.

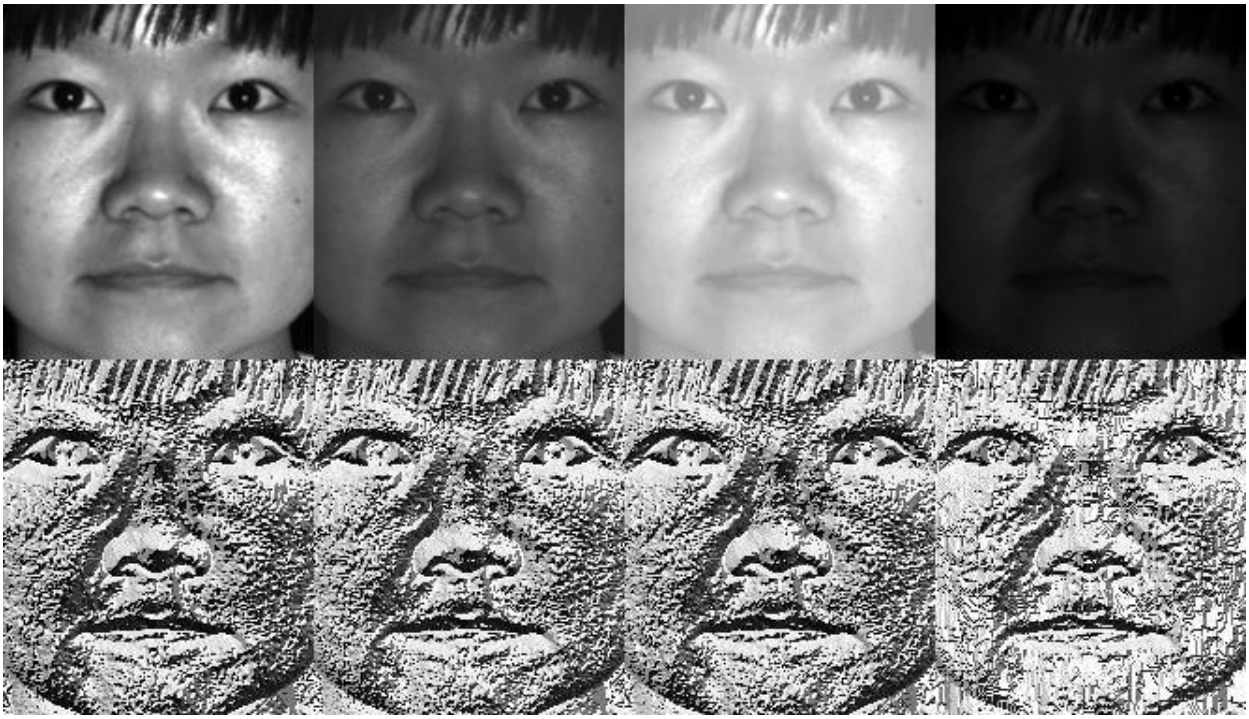

Fig. 10: Local Binary Histogram images

3.2.5.2 MATHEMATICAL EXPLANATION OF THE ALGORITHM

A more formal description of the LBP operator can be given as:

$$LBP(x_c, y_c) = \sum_{p=0}^{p-1} 2^p \, (i_p - i_c)$$

(x_c, y_c) is the central pixel with intensity i_c; and i_n being the intensity of the the neighbour pixel. s is the sign function defined as:

$$s(x) = \begin{cases} 1 & if\ x \geq 0 \\ 0 & else \end{cases}$$

This description captures very fine grained details in images. In fact, the authors were able to compete with state of the art results for texture classification. Soon after the operator was published it was noted, that a fixed neighbourhood fails to encode details differing in scale. So the operator was extended to use a variable neighbourhood. The idea is to align an arbitrary number of neighbours on a circle with a variable radius, which enables to capture the following neighbourhoods:

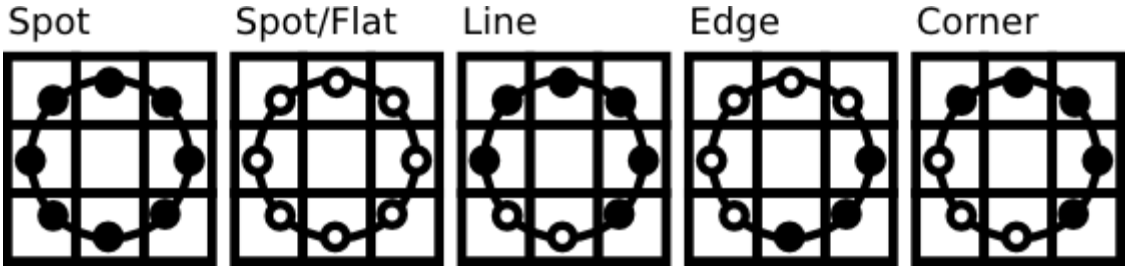

Fig. 11: Feature extraction using LBPH

For a given Point (x_c, y_c) the position of the neighbour (x_p, y_p) can be calculated by:

$$x_p = x_c + Rcos(\frac{2\pi p}{P})$$

$$y_p = y_c - Rsin(\frac{2\pi p}{P})$$

Where R is the radius of the circle and P is the number of sample points.

The operator is an extension to the original LBP codes, so it's sometimes called Extended LBP also referred to as Circular LBP. If a point's coordinate on the circle doesn't correspond to image coordinates, the point gets interpolated. Computer science has a bunch of clever interpolation schemes; the Open CV implementation does a bilinear interpolation:

$$f(x,y) \approx [1-x \;\; x] \begin{bmatrix} f(0,0) & f(0,1) \\ f(1,0) & f(1,1) \end{bmatrix}$$

By definition the LBP operator is robust against monotonic gray scale transformations. We can easily verify this by looking at the LBP image of an artificially modified image. The representation proposed by Ahonen et. al [AHP04] is to divide the LBP image into m local regions and extract a histogram from each. The spatially enhanced feature vector is then obtained by concatenating the local histograms (not merging them). These histograms are called Local Binary Patterns Histograms.

CHAPTER 4

OBJECT DETECTION AND RECOGNITION

4.1 INTRODUCTION

Object detection and tracking is one of the critical areas of research due to changes in motion of object and variation in scene size, occlusions, appearance variations, ego-motion and illumination changes. Object detection is performed to check existence of objects in video frame and to detect that object. Then detected object can be classified in various categories such as humans, vehicles and other moving objects [2].

Object Recognition can be divided into a sequence of steps which involves Detection and Tracking. Locating an object in successive frames of a video is called tracking.

The definition sounds straight forward but in computer vision and machine learning, tracking is a very broad term that encompasses conceptually similar but technically different ideas.

4.2 TRACKING VERSUS DETECTION

1. **Tracking is faster than Detection**:

   Usually tracking algorithms are faster than detection algorithms. The reason is simple. Tracking an object that was detected in the previous frame lets the system to know lots of information about the object such as location of object in the previous frame and the direction

and speed of its motion. So in the next frame, this information will be used to predict the location of the object in the next frame and do a small search around the expected location of the object to accurately locate the object. A good tracking algorithm will use all information it has about the object up to that point while a detection algorithm always starts from scratch.

Therefore, while designing an efficient system usually an object detection is run on every nth frame while the tracking algorithm is employed in the n-1 frames in between. It is true that tracking benefits from the extra information it has, but you can also lose track of an object when they go behind an obstacle for an extended period of time or if they move so fast that the tracking algorithm cannot catch up. It is also common for tracking algorithms to accumulate errors and the bounding box tracking the object slowly drifts away from the object it is tracking. To fix these problems with tracking algorithms, a detection algorithm is run every so often.

2. **Tracking can help when detection fails**:

When running a face detector on a video and the person's face get's occluded by an object, the face detector will most likely fail. A good tracking algorithm, on the other hand, will handle some level of occlusion. In the experimented results, the MultipleInstance Learning (MIL) tracker works well under occlusion.

4.3 OBJECT DETECTION METHODS

4.3.1 TEMPLATE MATCHING

The idea in template matching is to locate the in a given image. As the name suggests, we will need a template image, which is nothing but the object which needs to be detected in subsequent frames of the video. This method is one of the most basic versions of object detection. It simply slides the template image over the input image and compares the template and patch of input image under the template image. Technically, a 2D convolution is done bwtween the template image and the upcoming input frame. It returns a grayscale image, where each pixel denotes how much does the neighbourhood of that pixel match with template.

If input image is of size $(W\ x\ H)$ and template image is of size $(w\ x\ h)$, output image will have a size of $(W - w + 1,\ H - h + 1)$. Once you got the result, you can find where is the maximum/minimum value. Take it as the top-left corner of rectangle and take (w, h) as width and height of the rectangle. That rectangle is the region of template in the frame.

For exact object matches, with exact lighting/scale/angle, this can work great. An example where these conditions are usually met is just about any GUI on the computer. The buttons and such are always the same, so you can use template matching. But in practical cases, the lighting conditions/ scale/ angle cannot be the same every time and thus failing the need.

4.3.2 FEATURE BASED MATCHING

Feature based matching involves extracting the features from the query image and the object to be detected and then match them to accurately detect and recognize the object. We will be looking into a couple of feature

extractors namely SIFT and SURF and also a couple of feature based matchers Brute Force and Flann Based Matchers.

Feature Extractors

SIFT(Scale Invariant Feature Transform)

SURF(Speeded up robust Features)

4.3.2.1 Scale Invariant Feature Transform (SIFT)

Most detection techniques and feature extractors like Harris corner detectors are rotation-invariant, which means, even if the image is rotated, we can find the same corners. It is obvious because corners remain corners in rotated image also. But, this doesn't work with scaling. A corner may not be a corner if the image is scaled. For example, check Figure 12. A corner in a small image within a small window is flat when it is zoomed in the same window. So Harris corner is not scale invariant.

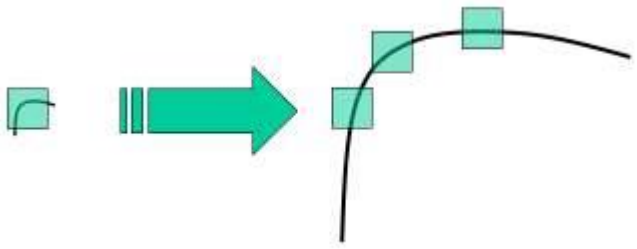

Fig 12: Detection of corners in an image

Later in 2004, D.Lowe, University of British Columbia, came up with a new algorithm, Scale Invariant Feature Transform (SIFT).

There are mainly four steps involved in SIFT algorithm:

1. **Scale space extrema detection**

From Figure 12, it is obvious that we can't use normal feature extractors to detect keypoints with different scale. To detect larger corners we need larger windows. For this, scale-space filtering is used. In it, Laplacian of Gaussian (LoG) is found for the image with various σ values. LoG acts as a blob detector which detects blobs in various sizes due to change in σ. In short, σ acts as a scaling parameter. For eg, gaussian kernel with low σ gives high value for small corner while guassian kernel with high σ fits well for larger corner. So, we can find the local maxima across the scale and space which gives us a list of (x, y, σ) values which means there is a potential keypoint at (x,y) at σ scale. But this LoG is a little costly, so SIFT algorithm uses Difference of Gaussians (DoG) which is an approximation of LoG. Difference of Gaussian is obtained as the difference of Gaussian blurring of an image with two different σ, let it be σ and $k\sigma$. This process is done for different octaves of the image in Gaussian Pyramid. It is represented in Fig13.

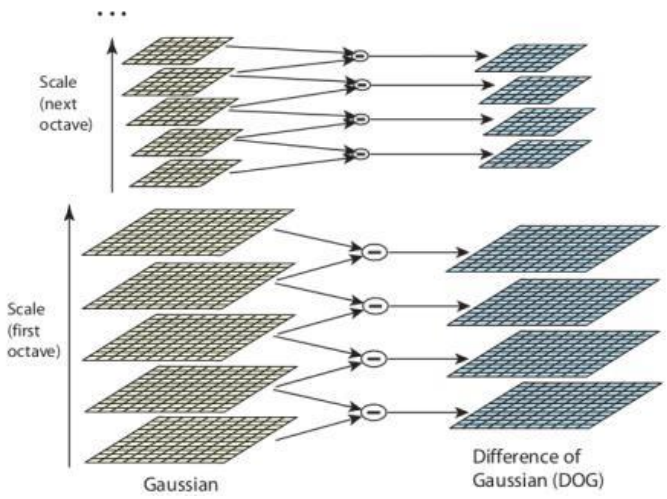

Fig 13: Scale space extrema detection

Once this DoG is found, images are searched for local extrema over scale and space. For eg, one pixel in an image is compared with its 8 neighbours as well as 9 pixels in next scale and 9 pixels in previous scales. If it is a local extrema, it is a potential keypoint. It basically means that keypoint is best represented in that scale. It is shown in Figure 14.

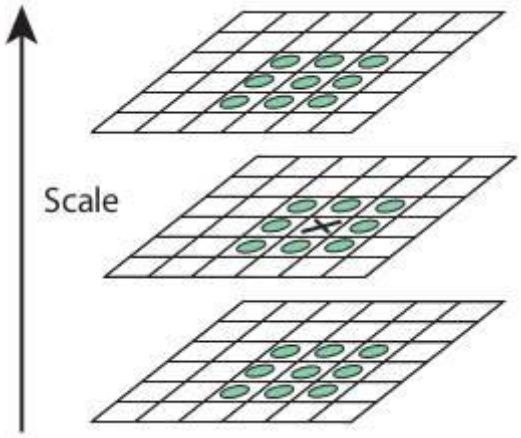

Fig 14: Potential keypoints

Regarding different parameters, the paper gives some empirical data which can be summarized as, number of octaves = 4, number of scale levels = 5, initial $\sigma = 1.6$, $k = \sqrt{2}$ etc as optimal values.

2. Keypoint localization

Once potential keypoints locations are found, they have to be refined to get more accurate results. Taylor series expansion of scale space is used to get more accurate location of extrema, and if the intensity at this extrema is less than a threshold value, which is generally 0.3, it is rejected. This threshold is called contrast Threshold in OpenCV.

DoG has higher response for edges, so edges are also removed. For this, a concept similar to Harris corner detector is used. A 2x2 Hessian matrix (H) to compute the pricipal curvature is used in this detector. We know from Harris corner detector that for edges, one eigen value is larger than the other. So here they used a simple function, if this ratio is greater than a threshold, called edge Threshold in, that keypoint is discarded. Basically, it eliminates any low-contrast keypoints and edge keypoints and what remains is strong interest points which can be used for object detection.

3. Orientation assignment

To achieve invariance to image rotation, now an orientation is assigned to each keypoint. A neigbourhood is taken around the keypoint location depending on the scale, and the gradient magnitude and direction is calculated in that region. An orientation histogram with 36 bins covering 360 degrees is created. It is weighted by gradient magnitude and gaussian-weighted circular window with σ equal to 1.5 times the scale of keypoint. The highest peak in the histogram is taken and any peak above 80% of it is also considered to calculate the orientation. It creates keypoints with same location and scale, but different directions. It contribute to stability of matching.

4. Keypoint descriptor

For a keypoint descriptor to be created, a 16x16 neighbourhood around the keypoint is taken. It is divided into 16 sub-blocks of 4x4 size. For each sub-block, 8 bin orientation histogram is created. So a total of 128 bin values are available. It is represented as a vector to form keypoint descriptor. In addition to this, several measures are taken to achieve robustness against illumination changes, rotation etc.

5. **Keypoint matcher**

Keypoints between two images are matched by identifying their nearest neighbours. But in some cases, the second closest-match may be very near to the first. It may happen due to noise or some other reasons. In that case, ratio of closest-distance to second-closest distance is taken. If it is greater than 0.8, they are rejected. It eliminaters around 90% of false matches while discards only 5% correct matches.

4.3.2.2 SPEEDED UP ROBUST FEATURES (SURF)

SIFT for keypoint detection and description was comparatively slow and people needed more speeded-up version. In 2006, three people, Bay, H., Tuytelaars, T. and Van Gool, L, published another paper, “SURF: Speeded Up Robust Features” which introduced a new algorithm called SURF. As name suggests, it is a speeded-up version of SIFT.

In SIFT, Lowe approximated Laplacian of Gaussian with Difference of Gaussian for finding scale-space. SURF goes a little further and approximates LoG with Box Filter. Figure 15 shows a demonstration of such an approximation. One big advantage of this approximation is that, convolution with box filter can be easily calculated with the help of integral images. And it can be done in parallel for different scales. Also the SURF rely on determinant of Hessian matrix for both scale and location.

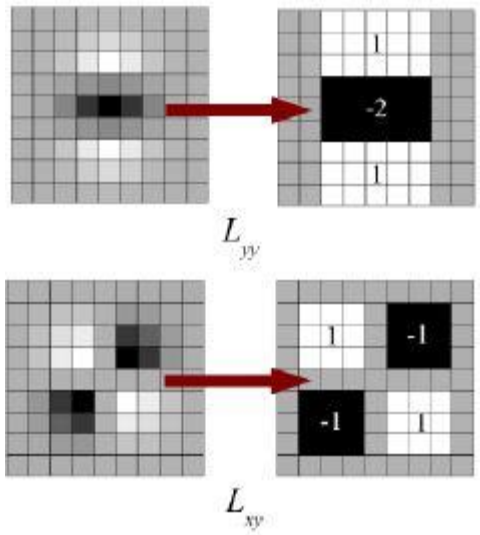

Fig 15: Approximation of DoG with LoG

For orientation assignment, SURF uses wavelet responses in horizontal and vertical direction for a neighbourhood of size 6s. Adequate guassian weights are also applied to it. Then they are plotted in a space as given in Fig 16. The dominant orientation is estimated by calculating the sum of all responses within a sliding orientation window of angle 60 degrees. Interesting thing is that, wavelet response can be found out using integral images very easily at any scale. For many applications, rotation invariance is not required, so no need of finding this orientation, which speeds up the process. SURF provides such a functionality called Upright-SURF or U-SURF. It improves speed and is robust upto $\pm 15^{\circ}$. OpenCV supports both, depending upon the flag, upright. If it is 0, orientation is calculated. If it is 1, orientation is not calculated and it is more faster.

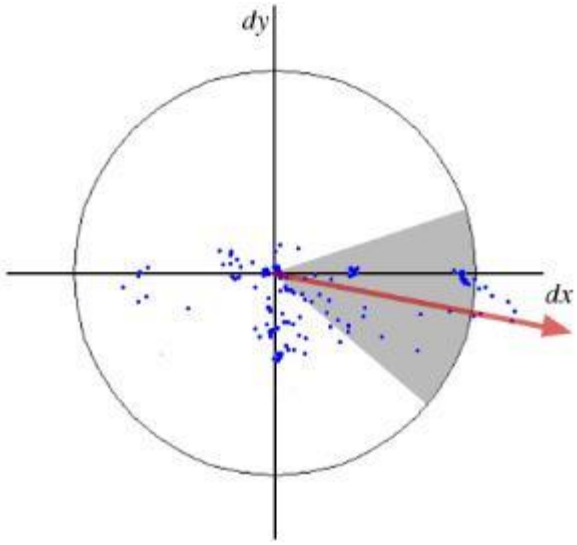

Fig 16: Plotting of gaussian weights

For feature description, SURF uses Wavelet responses in horizontal and vertical direction because use of integral images makes things easier. A neighbourhood of size 20sX20s is taken around the keypoint where s is the size. It is divided into 4x4 subregions. For each subregion, horizontal and vertical wavelet responses are taken and a vector is formed like this, $v = (\sum d_x \sum d_y \sum d_y \sum d)$. This when represented as a vector gives SURF feature descriptor with total 64 dimensions. Lower the dimension, higher the speed of computation and matching, but provide better distinctiveness of features.

For more distinctiveness, SURF feature descriptor has an extended 128 dimension version. The sums of $\mathrm{d_x}$ and $|\mathrm{d_x}|$ are computed separately for $\mathrm{d_y} < 0$ and $\mathrm{d_y} \geq 0$. Similarly, the sums of d_y and d_x are split up according to the sign of d_x , thereby doubling the number of features. It doesn't add much computation complexity. OpenCV supports both by setting the value of flag extended with 0 and 1 for 64-dim and 128-dim respectively (default is 128-dim)

Another important improvement is the use of sign of Laplacian (trace of Hessian Matrix) for underlying interest point. It adds no computation cost since it is already computed during detection. The sign of the Laplacian distinguishes bright blobs on dark backgrounds from the reverse situation. In the matching stage, we only compare features if they have the same type of contrast. This minimal information allows for faster matching, without reducing the descriptor's performance.

In short, SURF adds a lot of features to improve the speed in every step. Analysis shows it is 3 times faster than SIFT while performance is comparable to SIFT. SURF is good at handling images with blurring and rotation, but not good at handling viewpoint change and illumination change

4.4 OBJECT MATCHING TECHNIQUES

- Brute-ForceMatcher
- Flann – Bases Matcher

4.4.1 BRUTE – FORCE MATCHING TECHNIQUE

Brute-Force matcher takes the descriptor of one feature in first set and is matched with all other features in second set using some distance calculation. And the closest one is returned.

For BF matcher, first we have to create the BF Matcher object. It takes two optional params. First one is normType. It specifies the distance measurement to be used. By default, it is cv2.NORM_L2. It is good for SIFT, SURF etc.

For binary string based descriptors like ORB, BRIEF, BRISK

etc, cv2.NORM_HAMMING should be used, which used Hamming distance as measurement.

Second param is boolean variable, crossCheck which is false by default. If it is true, Matcher returns only those matches with value (i,j) such that i-th descriptor in set A has j-th descriptor in set B as the best match and vice-versa. That is, the two features in both sets should match each other. It provides consistant result, and is a good alternative to ratio test proposed by D.Lowe in SIFT paper.

Once this process is done, the next two importants methods to be done are BFMatcher.match() and BFMatcher.knnMatch(). First one returns the best match. Second method returns k best matches where k is specified by the user. It may be useful when we need to do additional work on that. It stacks two images horizontally and draw lines from first image to second image showing best matches. There is also cv2.drawMatchesKnn which draws all the k best matches. If k=2, it will draw two match-lines for each keypoint. So we have to pass a mask if we want to selectively draw it.

4.4.2 FLANN BASED MATCHING TECHNIQUE

FLANN stands for Fast Library for Approximate Nearest Neighbors. It contains a collection of algorithms optimized for fast nearest neighbor search in large datasets and for high dimensional features. It works more faster than BFMatcher for large datasets.

For FLANN based matcher, we need to pass two dictionaries which specifies the algorithm to be used, its related parameters etc. First one is IndexParams. For various algorithms, the information to be passed is explained in FLANN documentary. As a summary, for algorithms like SIFT, SURF etc, the following needs to be passed

```
index_params = dict(algorithm = FLANN_INDEX_KDTREE, trees = 5)
```

While using ORB, you can pass the following. The commented values are recommended as per the docs, but it didn't provide required results in some cases. Other values worked fine.

```
index_params= dict(algorithm = FLANN_INDEX_LSH,
                   table_number = 6, # 12
                   key_size = 12,     # 20
                   multi_probe_level = 1) #2
```

Second dictionary is the SearchParams. It specifies the number of times the trees in the index should be recursively traversed. Higher values gives better precision, but also takes more time.

CHAPTER 5

IMPLEMENTATION DETAILS

In this chapter, we describe the specifics of the problem, the dataset, and the algorithms that we used. We also detail the results that we have achieved.

5.1 PROBLEM FORMULATION

The system involves application of both object recognition and face recognition in online videos to enhance the viewing experience. In Object recognition, the idea is to learn the SIFT features from the object to be detected and then track the upcoming frames in the video for cross match of these features. In Face recognition, the LBPH face recognizer trains the computer to recognize faces in an input video frame. Web scraping and creating an interactive GUI are the other facets from its application perspective.

5.2 SOFTWARE PLATFORM EMPLOYED

5.2.1 PYTHON 3.5.4

Python is an interpreted high-level programming language for general-purpose programming created by Guido van Rossum and was first released in 1991. The design philosophy of Python emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library. The Python Package Index (PyPI) hosts thousands of third-party modules for Python. Both Python's standard library and the community-contributed modules allow for endless possibilities.

Python interpreters are available for many operating systems like Windows, IOS and Linux. C Python, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. C Python is managed by the non-profit Python Software Foundation. We have chosen Python for this project because of all these benefits.

5.2.2 OPEN CV 3.4.0

Open CV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. Open CV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. The library is used extensively in companies, research groups and by governmental bodies. The library contains more than 2500 optimized algorithms. These algorithms include a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. The user community extends over 47,000 people and the estimated number of downloads is over 14.

Well-established companies like Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, and Toyota employ this library in their products. Open CV’s

deployed uses, span the range from stitching street view images together, detecting intrusions in surveillance video in Israel, monitoring mine equipment in China, helping robots navigate and pick up objects at Willow Garage, detection of swimming pool drowning accidents in Europe, checking runways for debris in Turkey, inspecting labels on products in factories around the world on to rapid face detection in Japan.

Open CV has C++, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. It leans mostly towards real-time vision applications and takes advantage of MMX and SSE instructions when available. Open CV is written natively in C++ and has a template interface that works seamlessly with STL containers. That is why we have used it for all our video processing needs.

5.3 OBJECT RECOGNITION USING SIFT FEATURES

Template Matching is the most basic version of object detection. It simply slides the template image over the input image (as in 2D convolution). It, then, compares the template and patch of input image under the template image. It returns a grayscale image, where each pixel denotes how much does the neighbourhood of that pixel match with template. Maximum correlation or minimum difference means maximum matching. But, this method of detection works only if the object to be detected is not scaled or rotated. So, this does not work in real time detections as shown in Figure 17 as the object of interest (highlighted by yellow square) scales down gradually. Moreover, the other problems we observed were:

1. Computation was highly inefficient.
2. It is affline variant.

3. Even if object of interest was not present, it gave a match based on maximum threshold match.

So, in Figure 18 (a) and (b), the objects of interest were the mobile in the person's hand and the batman shirt he is wearing respectively. But, when template matching technique was used, his mobile and shirt were not detected if their orientations changed as shown in Figure 18 (b) and (c) respectively.

Fig. 17: Template matching in videos

Fig. 18 (a)

Fig. 18 (b)

Fig. 18 (c)

Fig. 18: Observations made from initially tried technique

Most detection techniques and feature extractors like Harris corner detectors are rotation-invariant, which means, even if the image is rotated, we can find the same corners. It is obvious because corners remain corners in the rotated image also. But, this doesn't work with scaling. So, in 2004, D.Lowe, University of British Columbia, came up with a new algorithm, Scale Invariant Feature Transform (SIFT) in his paper, Distinctive Image Features from Scale-Invariant Keypoints, which extract keypoints and compute its descriptors. This is what we have employed now in our project.

There are mainly four steps involved in SIFT algorithm.

1. **Scale –space extrema detection**
2. **Keypoint Localization**
3. **Orientation Assignment**
4. **Keypoint Descriptor**
5. **Keypoint Matching**

The SIFT Features of the given template can be found this way and stored in a variable. Now, we have to search for existence of these features in the every frame of the video. If a match is detected, the length and width of the template image is copied and these parameters are used to draw the rectange in the video frame containing the object.

Next step is to use Flann matcher to match the detected keypoints between the query frame and the template image. The Flann matcher takes the descriptor of one feature in first set and is matched with all other features in second set using distance calculation. And the closest one is returned. An example of Flann matching is shown in Figure 19.

Fig 19: Matching of SIFT features using Flann matcher

The next step is to detect the object with a polygon depending upon the number of matched features. If the detected features is greater than a preset threshold, the object exists in the image. The kepoints which are detected in the extreme corners are taken as coordinates and a polygon is drawn along the detected object as shown in Fig. 20.

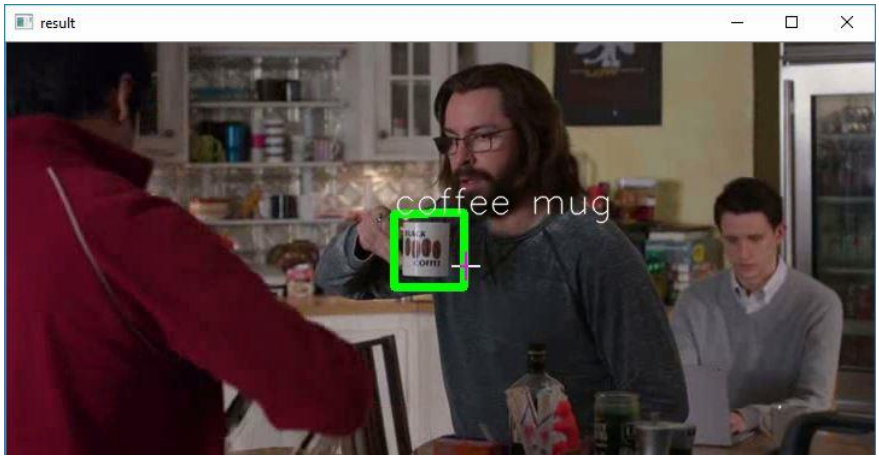

Figure 20: Detection of an object using SIFT features

Now, after the object is being detected, for easily buying the product which is advertised in a video, the concept of webscraping is used. A high level of interface to allow displaying web-based documents to users is provided. It will override the platform default list of browsers and accept the URL as the argument. Either window is raised or else new tab is opened, if not both an error is displayed with exception raised due to browser control error. Moreover, the GUI is built in a way that makes buying the product by means of a simple key press on the keyboard. Figure 21 shows the interactive GUI made for making this feasible. The "Product being advertised" statement rolls over on the bottom of the frame when the product is being advertised. The viewer if interested can pause the video to know more about the product and buy it instantaneously.

Fig. 21: Product advertised during playback

Upon pressing the space bar, more information about the product is displayed as shown in Figure 22.

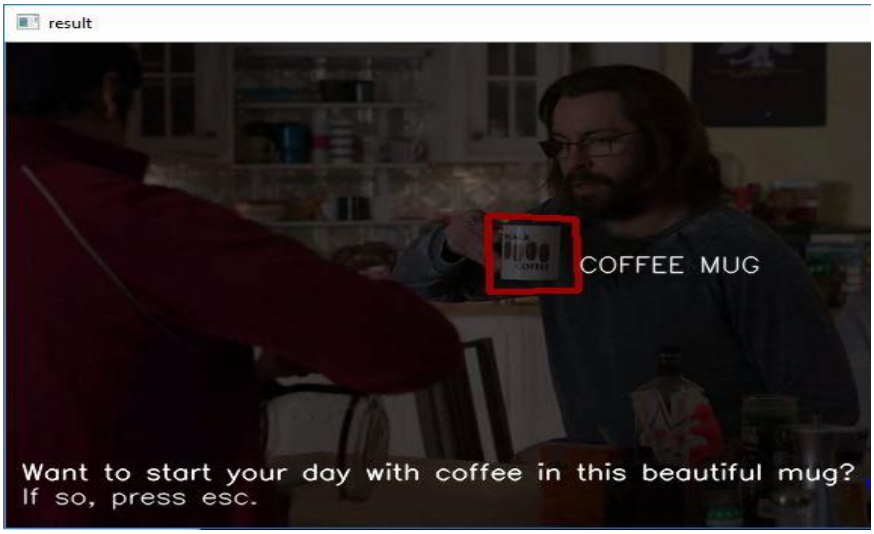

Fig. 22: Product information in detail

If the Escape key bar is pressed on the keyboard, the page redirects to the product page, where we can actually buy this product. Figure 23 shows the product page which pops up when the escape key is pressed.

Fig. 23: Redirected page to buy the product advertised

5.4 FACE DETECTION USING HARR FEATURES BASED CASCADE TRAINING

For a face to be recognized it needs to be detected first. By using Viola Jones algorithm, the system was trained to detect faces in any given frame. The steps used in training a Harr cascade file are listed below:

1) **Collecting Dataset:**

   First step is to grab positve images and negative images to train the cascade. Positive image is the cropped image of the face which needs to be detected in an image. Negative image is any image which doesn't contain the image which we want to detect. In our case, any image without a face is an negative image. Generally the number of positive images must be twice to that of negative images.

2) **Sample Creation:**

   The function written now super imposes the positive image on the negative images and creates multiple versions of positive images.

3) **Preprocessing:**

   The training is usually a very tedious process and can take several hours before our harr cascade file gets trained. To reduce the total time taken, certain pre-processing needs to be done. We scaled down the images to 50x50 and also converted every image to grayscale.

4) **Descriptor file for the dataset**

   The next step is to create the descriptor file also called as the background file. The file contains the properties of the image and also

the storage location of each image. In case of positive images, it also contains the coordinates of the face in the given image.

5) Training the cascade:

By using train cascade function in Open CV, the training will be completed in 7 to 8 hours time. Since this is just to detect any face, it is a one-time training for recognition of any face from any video.

opencv_traincascade -data data -vec positives.vec -bg bg.txt -numPos 1800 -numNeg 900 -numStages 10 -w 20 -h 20

Then, as shown above, we specified where we want the data to be stored, where the vector file is, where the background file is, number of positive images and negative images to be used, number of stages, and the width and height. Note that, we used significantly less numPos than we have. This is to make room for the stages, which will add to this.

Once the Harr cascade file is trained, it can be used to detect any face in a given image like in Fig. 24.

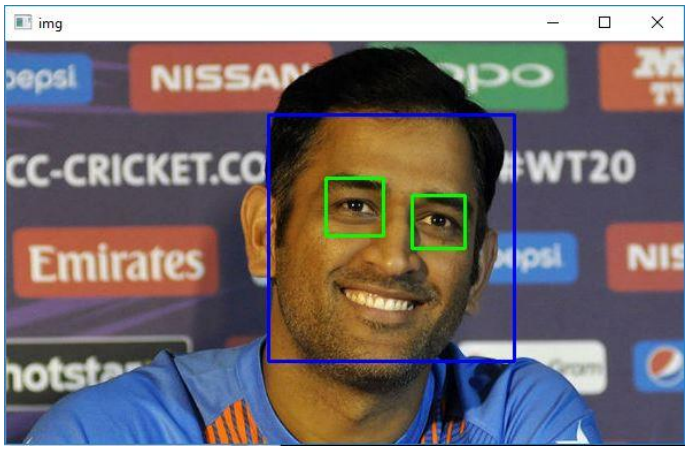

Fig. 24: Detection of Face in an image using Harr Cascade

5.5 FACE RECOGNTION USING LBPH RECOGNIZER

The steps involved include:

1) **Collection of Dataset**: The set of different faces, which need to be recognized, is collected first. We collected about 20 images of each person. We observed that: the more the number of images, the more will be the recognition accuracy.

2) **Pre-processing:** The collected images of people need to be scaled down for faster training. For this purpose, every image was scaled down to 50x50 pixels and was also converted to grayscale.

3) **Training the recognizer:** After the collection of dataset, they were used to train the recognizer.

Figure 25 shows the sample output after the person is recognized in a given video.

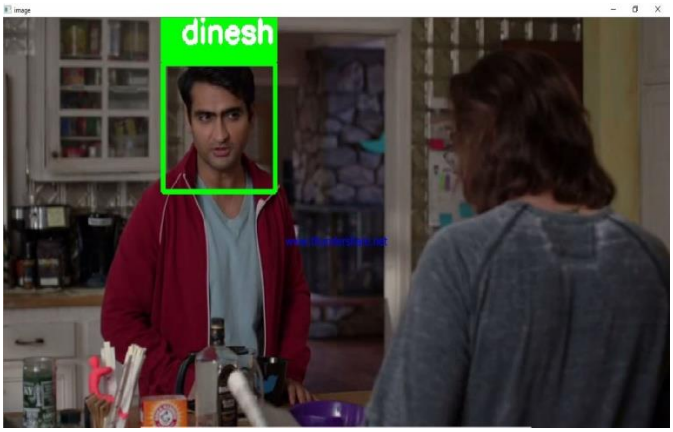

Fig. 25: Face Recognition using LBPH Recognizer

To eliminate the need for additional browsing and to get more information about the characters, face recognition is employed and information is

displayed about the character by clicking on the character's face. When the viewer wants to know about the actor in the particular scene, one can simply pause the screen. In the backend, the face recognition algorithm is run. Running this algorithm only when required saves unnecessary processing. When the screen is paused, face recognition is employed on the frame and the person's face id detected and recognized as shown in figure 26.

Fig. 26: Detection and recognition of a person's face

Now, to know more about the character detected as in data about that particular character and his/her bio-data, the viewer can simply click on the actor's face. Along with the information collected through web-scraping, the IMDb webpage link of that particular actor is displayed as shown in Fig. 27.

Fig 27: IMDb information about the actor

5.6 WEB – SCRAPING

The need and importance of extracting data from the web is becoming increasingly loud and clear. There are several ways to extract information from the web. Use of Application Program Interface (API's) is the best way to extract data from a website. Almost all large websites like Twitter, Facebook, Google, Twitter, and StackOverflow provide APIs to access their data in a more structured manner. API gives access to structured data from the provider, but other methods involve creation of an engine to extract the same information.

The main problem is, not all websites provide an API. Some do it because they do not want the readers to extract huge information in structured way, while others don't provide APIs due to lack of technical knowledge. In such cases, web scrapping is the best idea to extract information from the internet.

Web scraping is a computer software technique of extracting information from websites. This technique mostly focuses on the transformation of unstructured data (HTML format) on the web into structured data (database or spread sheet). There are two steps: one is fetching the URLs of the webpages and next is to extract information from the webpages. Functions and classes were defined to help with URL actions (basic and digest authentication, redirections, cookies, etc.). The library "Beautiful Soup" was then used to assist for pulling out information from a webpage.

In this project, we have used web scrapping for extracting the IMDb information about the actor/actress whenever the viewer clicks on the person’s face. In addition, web scrapping is also used to fetch the webpage when the viewer wants to buy the object of interest. The application of web scrapping in this project has been facilitated with an interactive GUI, which makes it convenient for the viewers.

CHAPTER 6

PERFORMANCE EVALUATION

The SIFT object detection algorithm used in this project extracts features in the given template to match it with the object's features in the frame of the video. Simple object detection algorithms like template matching do not consider features for detection, which makes it difficult for detecting objects at different lighting conditions and scaling. SIFT features extracted from the template and the video frame, along with the application of Flann based matcher, works perfectly for different lightning conditions and scaling variations. But, the problem associated with this is the fact that extracting SIFT features from an image is a slow process. This might make the video playback a bit slower than its usual speed. This is the first observation we made regarding the system's performance.

The face recognition algorithm used here is Local Binary Patterns Histogram (LPBH). This algorithm works better than Eigen faces and Fisher faces recognizers as they are illumination invariant. This algorithm is used efficiently in this project to recognize faces at any instant. This LPBH algorithm does not interfere with the speed of the video as the algorithm is executed only when the pause button is pressed. The main drawback in face recognition is that, it relies almost entirely on face detection. If the face detection fails at certain instants, the face recognition won't work at those instances as well. The Harr cascade algorithm is used for face detection and it detects only frontal face images, therefore this algorithm fails when the person is facing side wards, unless it is trained so. Except for those instances,

the face detection and face recognition algorithms work almost accurately with an efficiency of about 89 %.

CHAPTER 7

FUTURE SCOPE

The speed of detection and recognition is attributed to the speed of the computer. It can be made faster by working on the backend code to not get struck up between infinite loops and different decision cases. Moreover, the object detection algorithm runs for every single frame, thus making the video playback a bit slow. This needs to be fixed by, increasing the detection speed. The object which needs to be detected for advertising will appear in the video for a minimum of 5 seconds which would consist of more than 30 frames. This fact can be utilised to improve the video playback speed. The idea is to run the object detection algorithm for every 5 frames instead of checking for feature match in every frame.

The GUI is made in a way that it provides ease of access to everyone. But, this is in a prototype level. To make it as a product and to make it even more convenient for the users with additional features which any video player can provide, this system must be integrated with a modified video player. Instead, we can also use pre-existing players, which can run Python scripts with OpenCV library.

CHAPTER 8

CONCLUSION

The inconvenience faced by the viewers because of lengthy advertisements while watching their favorite shows has grown over the years. This situation has forced the users to use ad-blocking software in their personal computers, mobile phones and laptops. This ad- blocking softwares lock away all the advertisement content which leads to complete failure of the marketing schemes employed by the product based companies [19].

Most of the online viewers feel, advertisements are, too often poorly executed, annoying, code-heavy, and privacy-invading especially when they stream a video online or the News Feed on Facebook, without even considering their preferences. Programmatic ads, automatically placed by Google and others, are especially junky and repetitive. But, publishers have little control over them. Viewers go crazy trying to mute auto playing video advertisements on one of many tabs they have opened in a desktop browser. On some sites, native ads — in which advertiser-written articles or videos are intermingled with standard content — are too hard to distinguish from editorial matter. Clearly, the current method of advertising in online platforms is troublesome for the viewers and makes the user hate the product which gets advertised in-between without even having used it.

This project dealt with problems users face while watching videos and provided a smart platform for product manufactures to advertise their products. Clearly, the viewing experience of videos was enhanced by smart

advertising. Object recognition using SIFT Feature detection was employed on products which the advertiser wants to advertise. Web Scrapping of products website along with this algorithm facilitated buying the product which the user wants, by means of a simple key press.

In addition to this, the project also concentrated on eliminating the additional efforts required by the viewer to know about the characters in the video playback. LBPH Face recognition algorithm was implemented in this system to get IMDb information about the movie actor by simply clicking on the actor's face. To incorporate face recognition, object detection and webscraping for advertising and fetching IMDb data, an interactive Graphical User Interface was created, which makes viewing experience very comfortable for the user.

REFERENCES

1. Bai X. M., Yin B. C., Shi Q., and Sun Y. F., Face recognition using extended Fisherface with3D Morphable model, Proceedings of the Fourth International Conference on Machine Learning and Cybernetics, Guangzhou, August 2005, 18-21.

2. Belhumeur P. N., Hespanha J. P., and Kriegman D. J., Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection, IEEE Trans. Pattern Anal. Machine Intel, 19, May 1997, 711-720.

3. Mr. Deepjoy Das and Dr. Sarat Saharia," Implementation and Performance Evaluation of Background Subtraction Algorithms", International Journal on Computational Sciences & Applications (IJCSA) Vol.4, No.2, April 2014.

4. Duda R., and Hart P., Pattern Classification and Scene Analysis. New York: Wiley, 1973.

5. Er M.J., Wu S., Lu J., Toh H.L., Face recognition with radial basis function (RBF) neural networks, IEEE Trans. Neural Net,13, 2002, 697-710.

6. Girosi F., and Poggio T., Networks and the best approximation property, Biol. Cybern, (63), 1990, 169-176.

7. Himani S. Parekh, Darshak G. Thakore, Udesang K. Jaliya,"A Survey on Object Detection and Tracking Methods", International Journal of Innovative Research in Computer and Communication Engineering, Vol. 2, Issue 2, February 2014.

8. J.Joshan Athanesious, P.Suresh,"Systematic Survey on Object Tracking Methods in Video", International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), Volume 1, Issue 8, October 2012.

9. Kelly M. D., Visual Identification of People by Computer. Stanford AI Project, Stanford, CA, Technical Report, 1970, AI-130.

10. Kittikhun Meethongjan., and Dzulkifli Mohamad., A Summary of literature review: Face... (PDF Download Available).

Availablefrom:https://www.researchgate.net//publication/227709347_A_Summary_of_Face_Recognition_Literature_Survey [accessed Mar 31 2018].

11. Lu J., Yuan X., and Yahagi T., A method of Face recognition based on Fuzzy c-Means clutering and associated sub-NNs. Proc. IEEE, 18(1), 2007, 150-159.

12. Moody J., and Darken C.J., Fast learning in network of locallytuned processing units, Neural Computing, vol 1, page 281–294, 1989.

13. Navneet Dalal, Bill Triggs, and Cordelia Schmid," Human Detection Using Oriented Histograms of Flow and Appearance", April 2006.

14. Qiang Zhu, Shai Avidan, Mei-Chen Yeh, Kwang-Ting Cheng," Fast Human Detection Using a Cascade of Histograms of Oriented Gradients", June 2006.

15. Khushboo Khurana, Reetu Awasthi, "Techniques for Object Recognition in Images and Multi-Object Detection", International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 2, Issue 4, April 2013.

16. Rowley H., Baluja S., and Kanade T., Neural network-based face detection, IEEE Trans. Pattern Anal. Mach. Intell, 20, 1998, 23-38.

17. Seema Kumari, Manpreet Kaur, Birmohan Singh," Detection And Tracking Of Moving Object In Visual Surveillance System", International Journal of Advanced Research in Electrical Electronics and Instrumentation Engineering (IJAREEIE) Vol. 2, Issue 8, August 2013.

18. Sing J. K., Basu D. K., Nasipuri M., and Kandu M., Face recognition using point symmetry distance-based RBF network, Applied soft computing, available at www.sciencedirect.com, l 7, January 2007, 58-70.

19. The Verge. (2018). Mossberg: Lousy ads are ruining the online experience.[online]Availableat:https://www.theverge.com/2017/1/18/14304276/walt-mossberg-online-ads-bad-business [Accessed 3 Dec. 2017].

20. Turk M. and Pentland A., Eigenfaces for Recognition, J. Cognitive Neuroscience, 3(1), 1991, 71-86.

21. Valentin D., Abdi H., O'Toole A.J., and Cottrell G.W., Connectionist models for face processing: a survey, Pattern Recognition, 27, (1994), 1209–1230.

22. Yang F., Paindovoine M., Implementation of an RBF neural network on embedded systems: real-time face tracking and identity verification, IEEE Trans. Neural Network, (14), 2003, 1162-11