BACHELOR OF SCIENCE IN
COMPUTER SCIENCE AND ENGINEERING

Inspiring Excellence

Machine Learning Approach for Face
Recognition from 3D Models Generated By
Multiple 2D Angular Images

Authors

Moinuddin Mahmud (14301119)

Shegufta Mehzabin (14201013)

Sabrina Jahan Prova (14201011)

Supervisor

Dr. Md. Ashrafal Alam

Assistant Professor

Department of CSE

A thesis submitted to the Department of CSE in partial fulfillment of the
requirements for the degree of B.Sc. Engineering in CSE

Department of Computer Science and Engineering
BRAC University, Dhaka - 1212, Bangladesh

December 2018

We would like to dedicate this thesis to our loving parents.

Declaration

It is hereby declared that this thesis/project report or any part of it has not been submitted elsewhere for the award of any Degree or Diploma.

Authors:

Moinuddin Mahmud
Student ID: 14301119

Shegufta Mehzabin
Student ID: 14201013

Sabrina Jahan Prova
Student ID: 14201011

Supervisor:

Dr. Md. Ashraful Alam
Assistant Professor, Department of Computer Science and
Engineering BRAC University

December 2018

Acknowledgements

Our heartiest thanks to our supervisor Dr. Md. Ashraful Alam. His help, support and guidance has brought us to the concluding success of our research.

We are very much grateful to our parents and to all our family members for their constant support and encouragement that worked as the source of our spirit and energy.

We will sincerely thank to all the technicians of our computer laboratories. Finally our warm gratefulness to our beloved BRAC University for giving us all these opportunities to work on the field that we have cherished to work on.

Abstract

We propose machine learning approach for face recognition from 3D models generated by multiple 2D angular images that recognizes faces from multiple angle of a 3D face model. Though, many works on identifying faces from 3D have already been done, there are many spaces to update, improve and contribute more features on previously done researches. However, this research includes SFM algorithm which is a combination of SIFT detector, Approximate Nearest Neighbors (ANN) algorithm and RANSAC algorithm to reconstruct 3D from multiple RGB images. Again, it includes AdaBoost Learning algorithm which was used to train model to recognize faces. Besides, we used Local Binary Pattern Histogram (LBPH) which is an effective texture administrator, marks the pixels of a picture by thresholding the area of every pixel. Finally, the System successfully recognizes faces which are deviated up to 60° angular deviation respectively to left and right (total: 120°). Additionally, it gives an accuracy of 80% to 100% depending on angular deviation of up to from 0° to 60° . Nevertheless, the rate of accuracy of our proposed system is reversely proportional to the Angular Deviation.

Table of contents

List of figures

List of tables

1	Introduction	1
1.1	Introduction	1
1.2	Motivation	2
1.3	Thesis Overview	3
2	Literature Review	5
3	Proposed Method	9
3.1	3D Reconstruction	9
3.1.1	SFM based 3D Modeling	9
3.1.2	Fundamentals of Structure from Motion	11
3.1.3	Structure from Motion Algorithm	13
3.1.4	Feature Detection	13
3.1.5	Scale-invariant Feature Transform (SIFT)	15
3.1.6	Approximate Nearest Neighbors (ANN) Algorithm	19
3.1.7	Matching Features	21
3.1.8	GUI Application	23
3.2	Data Training	25
3.2.1	Dataset Creator Algorithm	25
3.2.2	Training Dataset using HAAR Cascade Algorithm	27
3.2.3	Haar Feature Selection	27
3.2.4	Integral Image	29
3.2.5	The Local Binary Patterns (LBP)	31
3.2.6	AdaBoost Learning Algorithm	33
3.2.7	Cascade Classifier	38

3.3	Face Recognition	42
4	Result Analysis	47
4.1	Comparative Study and Accuracy	47
5	Conclusion	51
5.1	Difficulties	51
5.2	Future Plan	51
	References	53

List of figures

3.1	SFM Algorithm for Camera Parameter estimation	10
3.2	Flowchart of SFM Algorithm	12
3.3	Feature Extraction, Colored Arrows indicates features.	14
3.4	Feature extracted from multiple images: Arrows indicates features	18
3.5	p_j is the nearest point to q . In the c-approx nearest neighbor problem any point within the radius c_r is accepted.	19
3.6	Feature Extraction, Colored Arrows indicates features.	22
3.7	Test data: From top left : (a) Snapshot from 0 degree angle, (b) Snapshot from 30 degree (left Rotation) angle, (c) Snapshot from 60 degree (left Rotation)angle, (d)Snapshot from 90 degree (left Rotation) angle From bottom left: (e) Snapshot from 30 degree(right Rotation) angle, (f) Snapshot from 60 degree(right Rotation) angle and (g) Snapshot from 90 degree(right Rotation) angle	25
3.8	Dataset Creator Algorithm to create Grayscale Face DataSet.	26
3.9	Workflow of Dataset Creator Algorithm.	26
3.10	Rectangular Feature.	28
3.11	Integral Image blocks	29
3.12	Integral Image Sum	30
3.13	Integral Image: illustration example	30
3.14	Example of primitives detected by LBP	31
3.15	LBP calculation Example	32
3.16	Cascade Classifier	39
3.17	workflow of LBP operator	44
3.18	Selected neighbors point and radius from the central value. a) a)contain 8 neighbors and radius is 2, b)contain 8 neighbors and radius is 1, c) contain 12 neighbors and radius is 2, d) contain 12 neighbors and radius is 3	45

List of tables

4.1	Experimental results of few of the discussed 3D and 2D methods	48
4.2	Experiments & Result Analysis	49

Chapter 1

Introduction

1.1 Introduction

The humans can perceive the 3D shape of a 2D image by just looking at it, even if the object in the image is completely new to the eye. The human brain plays a vital role in obtaining this 3D world through 2D images. After noticing the 2D image, the human eye signals the brain about the object through a nerve signal. After processing the nerve signal the brain creates the 3D shape of the 2D object. Appearance of the object, familiarity with shapes of similar 3D objects and other similar factors assist in creating the aforementioned 3D shape. Though this is an unconscious act for the humans, when tried to simulate with computers, efficient and effective ways have to be explored for identifying object features to assist the reconstruction of the 3D face. Thus it makes the area of 3D shape reconstruction from 2D images a complex and a problematic one. The topic, 3D face reconstruction from 2D images has been derived and studied separately from the more general area of 3D shape reconstruction due to its depth and the complexity. Techniques for attaining facial information for 3D reconstruction are broadly categorized into three, namely, pure image-based techniques, hybrid image-based techniques and 3D scanning techniques. The pure image-based techniques perform the reconstruction using only 2D images without estimating the real 3D structure. In hybrid image based techniques both approximations and the data gained from images are used in the reconstruction process. The 3D scanning techniques have the capability to capture the complete 3D structure since scanned images provide both geometry and texture information of the face. Our face bears our initial identity. Face recognition is the process by which we identify a person. Face recognition framework is one of those innovation wonders where a machine can act nearly as human and recognize a person from his face. Nowadays we use face recognition in ATM and banks for identifying frauds. Face recognition is also used for identifying fraud voters. To keep records of students or

members of any groups face recognition is being used all over the world. Despite the fact that exploration in face recognition has crossed over three decades, most of the works were done for recognizing faces from 2D images. Only a few works were done in the field of 3D images. Three-dimensional human facial surface data is an amazing biometric methodology that can possibly enhance the identity and confirmation exactness of face recognition frameworks under many circumstances. In our whole thesis we worked to make all these cases stronger. Though, some researches on this field have already been done, we tried to append more features and improve earlier works.

Nowadays security issue has become a global issue. Again, All the Systems nowadays are using security system which includes biometric identity. In this case, face recognition from 3D model is a very interesting sector to work on. Nevertheless, 2D image data stored in Banks, offices, Schools and in several places which we are using every day are going to turn into 3D images as the technology advancing to the future. If that happens, face recognition from 3D model would be needed all over the world in upcoming days. In fact, Developed countries like USA, Russia are already using 3D models. So, it is a great opportunity to work on this field.

1.2 Motivation

We tried in our thesis to reconstruct a 3D model from 2D face images and recognize that from multiple angle. In this era of image processing, face recognition is one of the most popular field for people who are interested in doing researches. Multiple of works and thesis have been done on the topic of face recognition. However, for many decades it was confined in 2D images but in last few years this process is being applied for 3D images as it gives more accuracy in safety purpose. More the world is getting digitalized more the security system needs to get updated. The development of electronic exchange results in incredible interest for quick and exact user identification and confirmation. This was the main issue that motivated us to have our research on this particular.

Again, overcome security issues is another reason that drove us to 3D face recognition. Systems are advancing and all the system will turn into a system that would work with 3D data. Works are done with 2D images, but we wanted to see with a different angle. We wanted to see what happen if the data we are dealing will go to next steps as the technology advances. These are the issues that motivated us to work on this field.

1.3 Thesis Overview

Here in our research we started our job by taking multiple 2D pictures of our sample models and then reconstruct a 3D image of each model by a software which working process is based on Structure from Motion (SFM) algorithm. Then we trained our dataset using AdaBoost learning algorithm. After that we did the part of recognition of faces from multiple angle from our 3D models. We tried to make a good rate of accuracy of our recognition process. In chapter 2 we discussed on fundamentals of face recognition. The main theme of our research is face recognition and we tried to present a constructive discussion of face recognition which is very important to catch the basic of our research. After that in chapter 3 we stated our proposed model which includes three sections of our work flows which are 3D reconstruction, data training and face recognition. In 3D reconstruction section we described the process and algorithm that has been used for 3D reconstruction using multiple 2D images. Then in second section we discussed on the process of training our data set through Ada Boost algorithm. Again in last step we stated about our final process that is recognizing or matching faces from 3D images. We included the accuracy rate of our system that we got by analyzing our recognition ability of our system from our training data set. We concluded our paper with some of our plans that we have got to implement in our project in future. Though it also has got some limitations we tried our best to make our work accurate and presentable.

Chapter 2

Literature Review

Faces, with all their complexities and vast number of degrees of freedom, allow us to communicate and express ourselves through expressions, mimics, and gestures. Human face is difficult to demonstrate notwithstanding utilizing ordinary 3D modeling programming; henceforth the task of remaking them according to features picked up from 2D pictures and making them sensible and without uncertainty, significantly more detailed. Reconstructing the detailed geometric structure of a face from a given image is a key to many computer vision and graphics applications. We are introducing 3D face recognition from multiple 2D images by using SFM algorithm which basically consists of SIFT, ANN and RANSAC algorithm which we used to reconstruct and get the accuracy value by comparing different input images having angular deviations. There are many papers, surveys on this topic but combining with accuracy is not published yet. We went through various papers where different approaches are used to solve this reconstruction problem but most have drawbacks and limitations. SFM based 3D modeling, feature extraction and linear matching of images are the main methods of attaining the facial information for reconstruction of its 3D counterpart. Though this topic has gained a lot of importance and popularity, a fully accurate facial reconstruction mechanism has not yet been identified due to the complexity and ambiguity involved. This paper discusses about the 3D face reconstruction using SFM algorithm and accuracy of images. It concludes with an analysis of several implementations and some accuracy speculations of 3D face reconstruction. In the paper "Automatic 3d face reconstruction based on single 2d image", the approach of Face recognition combines a number of advantages as it offers speed and reliability of detection at distance from the subject, which makes its candidacy as a biometric feature a fertile area of pattern recognition and image processing research. In general, changes to head pose and facial expression cause a 2D image of a face to become an unreliable form of identification, while a 3D representation suggests all vertices of the 3D facial model as candidate and useful features for face classification and

recognition [1]. Besides, in “3d face reconstruction from 2d images”, Package alteration is completed to decide more precise 3D camera positions, picture sets are redressed, and thick 3D confront data is removed without utilizing the earlier face information. Exceptions are evacuated by utilizing tensor casting a ballot. A 3D surface is removed from the thick 3D data and surface detail is extricated from the pictures [2]. To manage anomalies in both the meager and thick coordinating stages, past work either depends on a solid earlier model for face geometry or forces limitations on the camera movement. Furthermore, in “Model-assisted 3d face reconstruction from video”, Solid earlier models result in a genuine trade off in conclusive reproduction quality and commonly look somewhat like a conventional or mean face. Display based strategies, while giving the presence of face detail, in truth convey this detail over from the model earlier. Face highlights, for example, facial hair, moles, and other trademark geometry are lost. Movement confinements, for example, permitting just unadulterated pivot are about difficult to fulfill by the end client, particularly with a handheld camera [3]. Moreover, In the paper “Efficient 3D reconstruction for face recognition”, Face acknowledgment with variation posture, brightening and appearance (PIE) is a testing issue. In this paper, authors propose an investigation by-union system for face acknowledgment with variation PIE. Initial, a productive two-dimensional (2D)-to-three-dimensional(3D) incorporated face remaking approach is acquainted with recreate a customized 3D confront display from a solitary frontal face picture with unbiased demeanor and ordinary enlightenment. At that point, sensible virtual countenances with various PIE are combined dependent on the customized 3D face to describe the face subspace. At last, confront acknowledgment is led dependent on these agent virtual appearances. The broad trial results demonstrate that the orchestrated virtual faces essentially enhance the exactness of face acknowledgment with evolving PIE [4]. In “3D and infrared face reconstruction from RGB data using canonical correlation analysis”, CCA is a factor investigation technique which misuses the relationship between’s two high dimensional signs. Authors first utilize CCA to perform 3D confront recreation and in a different application and they anticipate close infrared (NIR) confront surface. In the two cases, the information are shading (RGB) confront pictures. Examinations appear, that because of the connection among’s info and yield flag, just few accepted variables are expected to portray the practical connection of RGB pictures to the separate yield with sensible precision [5]. Moreover, in “Combining High Resolution Color and Depth Images for Dense 3D Reconstruction”, authors depicts an effective method to reconstruct a dense 3D model of an object by combining high resolution color and depth images. Multiple views of color images can be used for reconstructing a 3D model of the captured object. Besides, a depth camera is capable to capture 3D distance information even in homogenous regions and it gives low resolution and is unable to provide

an accurate result of a detailed object. Authors also propose a combined method of utilizing both high resolution color and depth images to obtain a high-quality, accurate and dense 3D model [8]. In paper “3D Digitization using Structure from Motion”, computational methods is used to generate three dimensional models from photographs have received increasing attention in the last few years. In present days, these methods are not a substitutive to laser scanners, they are a viable alternative in cases were using scanner is not possible. These methods are also attractive because they do not need the acquisition of expensive materials. Besides, authors focused on the structure from motion approach, giving a brief explanation of theory, explaining some basic algorithms and the installation of Visual structure from Motion software [9]. In paper “Approximate Nearest Neighbors: Towards Removing the Curse of Dimension”, authors describes a method for reducing complexity owing to dimensionality is to project the points into a random sub-space of lower dimension but this would lead to a sub linear query time only for large values. Authors also introduce the notion of locality-sensitive hashing and apply it to sub linear-time similarity searching and PLEB procedures used in cases where points are being inserted and deleted over time [11]. Furthermore, In “ Rapid object detection using a Boosted Cascade of Simple Features”, describes a machine learning approach for visual object detection which is capable of processing images extremely rapidly and achieving high detection rates. Here, a learning algorithm based on AdaBoost which selects a small number of critical visual features from a larger set and yields extremely efficient classifiers. Their contribution is a method for combining increasingly more complex classifiers in a “cascade” which allows background regions of the image to be quickly discarded while spending more computation on promising object-like regions. The cascade can be viewed as an object specific focus-of-attention mechanism which unlike previous approaches provides statistical guarantees that discarded regions are unlikely to contain the object of interest. In the domain of face detection the system yields detection rates comparable to the best previous systems. Used in real-time applications, the detector runs at 15 frames per second without resorting to image differencing or skin color detection [17]. Besides, in “ Multiple Object detection using OpenCV on an Embedded Platform”, authors depicts object detection technology which has driven by an increasing processing power available in software and hardware. In this paper, they present a developed application for multiple objects detection based on OpenCV libraries. The complexity-related aspects that were considered in the object detection using cascade classifier .Moreover, the profiling and porting of the application into an embedded platform and comparing the results with the regular platform [19]. In paper “Robust Real-Time Face Detection”, describes a face detection framework that is capable of processing images extremely rapidly while achieving high detection rates. There are three key contributions. The first is the introduction of a

new image representation called the “Integral Image” which allows the features used by our detector to be computed very quickly. The second is a simple and efficient classifier which is built using the AdaBoost learning algorithm to select a small number of critical visual features from a very large set of potential features. The third contribution is a method for combining classifiers in a “cascade” which allows background regions of the image to be quickly discarded while spending more computation on promising face-like regions. A set of experiments in the domain of face detection is presented. The system yields face detection performance comparable to the best previous systems that is implemented on a conventional desktop, face detection proceeds at 15 frames per second [20]. Lastly, in paper “2D and 3D Face Recognition: A Survey”, represents a valid alternative of biometrics but they suffer of drawbacks and limitations. Iris scanning, for example, is very reliable but too intrusive; fingerprints are socially accepted, but not applicable to non-consentient people. On the other hand, face recognition represents a good compromise between what’s socially acceptable and what’s reliable, even when operating under controlled conditions. In last decade, many algorithms based on linear/nonlinear methods, neural networks, wavelets, etc. have been proposed. Nevertheless, Face Recognition of these approaches encountered problems in outdoor conditions. This lowered their reliability compared to state of the art biometrics. This paper provides an “ex cursus” of recent face recognition research trends in 2D imagery and 3D model based algorithms. To simplify comparisons across different approaches, tables containing different collection of parameters (such as input size, recognition rate, number of addressed problems) are provided. This paper concludes by proposing possible future directions [29].

Chapter 3

Proposed Method

3.1 3D Reconstruction

3.1.1 SFM based 3D Modeling

In this paper, a framework is intended for three dimensional demonstrate remaking. The proposed framework utilizes a camera to take numerous pictures of Subject and investigate those pictures utilizing structure from motion (SFM) to acquire the camera parameters and meager point mists. Besides, this framework can work with no camera adjustment before the tests. With the end goal to remake the objective protest, the fix based multi-see stereo is utilized to catch two target acts and process the three-dimensional point mists and shading data from the objective protest. At that point utilize the iterative nearest guide calculation toward enlist those two target stances. In the last advance, the Poisson surfaces recreation procedure is used to construct the three-dimensional work show. At last, the trial results demonstrate that the examination secures legitimate practicability and strength. [6].

We will describe our whole work flow which is actually in three steps. In our first step we took multiple 2D images as input and generated 3D image from those. Here we use an effective method to reconstruct a 3D image by combining multi view 2D face images. We utilize SFM algorithm to estimate camera parameter automatically.

Fig. 3.1 SFM Algorithm for Camera Parameter estimation

Figure 3.1 shows the diagram of the SFM algorithm for camera parameter estimation. The process is designed with multiple steps.

First, image feature points are detected by the SIFT keypoint detector. Second, the keypoint descriptors between the image pair are matched by an approximate nearest neighbors (ANN) algorithm. The fundamental matrix of the image pair is then robustly estimated by a random sample consensus (RANSAC) algorithm whereas the outlier matches are removed. After we estimate the fundamental matrix and find geometrically consistent matches for each pair, the matches are organized into tracks. Each track contains a connected set of matching keypoints across multiple images. Finally, a set of camera parameters and 3D structures are

jointly recovered by a bundle adjustment. The objective is to recover the parameters that minimize the re-projection error, i.e. a sum of distances between the projections of each track and its corresponding image features. Instead of heavily optimizing all cameras at the same time, we first select and estimate the parameters of the most reliable pair of cameras and add it one by one incrementally. It is more robust and efficient. The estimated camera parameter is then used for the 3D dense reconstruction.

3.1.2 Fundamentals of Structure from Motion

Structure from motion (SFM) is a standout amongst the most prevalent issues that analysts keen on the field of PC vision and PC illustrations. Structure from motion is the issue of recreating 3D from 2D pictures. As indicated by the information they utilize are an arrangement of highlights matches or an optical stream field, 3D recreation Approaches can be delegated include based or stream based techniques. Feature based recreation is completed utilizing comparing highlights in sets of pictures of a similar scene taken from various perspectives. In stream based Matching is supplanted by optical stream the highlights speed field created by the camera movement. In this paper, we talk about and dissect the endeavors of researchers in this field to cover all parts of structure from motion [7].

Structure from motion (SFM) algorithm basically deals with two problems. One is the studying an obscure structure from certain camera positions and another is getting and fixing camera motion from declared fix points. The connection between estimation of the different view geometry and the component following comprises of the numerous view connections being utilized to regularize the element following. The 3D structure of the separated attributes estimation is dependent on the attributes and gauge of the camera parameters. In the event that the removed attributes are image focuses, at that point the evaluated 3D structure is a 3D point cloud [2]. In the era of computer vision and visual recognition, structure from motion (SFM) alludes to the way toward finding the three-dimensional structure of a protest by dissecting neighborhood movement motions after some time. Structure from motion creates the 3D demonstrate from the movement sign. Accepting all protest in the scene are static, its relative dislodging depends just on their profundity. This procedure creates the model from an arrangement of photos or a video. All the more definitely, SFM frameworks take as info an arrangement of pictures and create a 3D point billow of the scene, the situation from which every photograph has been taken, and optical parameters of the cameras. It demonstrates the position an introduction of the cameras used to catch the scene, this data has been registered by the SFM calculation [8].

SFM Algorithm Pipeline

Fig. 3.2 Flowchart of SFM Algorithm

3.1.3 Structure from Motion Algorithm

Discovering structure from motion shows a comparable issue as discovering structure from stereo vision. Initial phase in the calculation is ordinarily to discover correspondences between pictures. With the end goal to discover these components, highlights, for example, corner directs require toward be found and followed starting with one picture then onto the next. Features of a picture are important regions of the picture that stay stable in the following picture, for example, corners (edges with angles in various ways). The component directions after some time are then used to recreate their 3D positions and the camera's movement. This assignment is testing in light of the fact that the picture development process isn't for the most part invertible, subsequently extra data is expected to explain the recreation. One probability is to misuse earlier information about the scene to diminish the quantity of degrees of opportunity. Another probability is to utilize relating picture indicates in various perspectives separate 3D data. In this paper we center the last choice to complete a nonexclusive 3D recreation from any sort of pictures without earlier learning. [9]

3.1.4 Feature Detection

In computer vision and images handling, the idea of attributes is utilized to mean a snippet of data which is important for illuminating the computational assignment identified with a specific application. There are different calculations for highlight discovery, yet the most well-known are which depend on Scale-invariant feature transform (SIFT) [9].

Fig. 3.3 Feature Extraction, Colored Arrows indicates features.

Figure 3.3 shows the Feature being extracted from the images. For this feature detection we used SIFT keypoint detector.

3.1.5 Scale-invariant Feature Transform (SIFT)

It is a feature detecting algorithm in interdisciplinary field of computer to calculate and state local attributes of images. It is a procedure for identifying striking, stable element focuses in an image. For each such point, it likewise gives an arrangement of features that portrays a little picture area around the point. These highlights are invariant to pivot and scale.

In first step it is need to locate approximate key points and determine the size of striking element focuses. Through differences of Gaussians the density of changes can be observed. Then those scale and location should be refined. Then again it is needed to calculate the statements of each key points.

Scale-space Extrema Detection

We start by distinguishing purposes of intrigue, which are named keypoints in the SIFT structure. The picture is convolved with Gaussian channels at various scales, and after that difference of successive Gaussian-blurred images are taken. We took Keypoints as maxima/minima of the Difference of Gaussians (DOG) that happen at different scales. In particular, a DOG picture $D(x, y, \sigma)$ [40] is given by

$$D(x, y, \sigma) = L(x, y, k_i\sigma) - L(x, y, k_j\sigma) \quad (3.1)$$

Here the convolution of the original image $I(x,y)$ is $L(x,y,k,\sigma)$ with the Gaussian blur $G(x,y,k\sigma)$ at the scale of $k\sigma$ [40].

$$L(x, y, k\sigma) = G(x, y, k\sigma) \times I(x, y) \quad (3.2)$$

Consequently a DoG picture between scales $k_i\sigma$ and $k_j\sigma$ is only the distinction of the Gaussian-Blurred pictures at scales $k_i\sigma$ and $k_j\sigma$. For scale space extrema discovery in the SIFT calculation, the picture is first convolved with Gaussian-blur at various scales. The convolved pictures are assembled by octave (an octave compares to multiplying the estimation of σ , and the estimation of k_i is chosen with the goal that we acquire a settled number of convolved pictures per octave. At that point the Difference-of-Gaussian pictures are taken from nearby Gaussian-blur pictures per octave.

When DOG pictures have been acquired, keypoints are recognized as neighborhood minima/maxima of the DOG pictures crosswise over scales. This is finished by looking at every pixel in the DOG pictures to its eight neighbors at a similar scale and nine comparing neighboring pixels in every one of the neighboring scales. In the event that the pixel esteem is the greatest or least among all thought about pixels, it is chosen as an applicant keypoint.

This keypoint recognition step is a variety of one of the mass identification techniques created by Lindeberg by distinguishing scale-space extrema of the scale standardized Laplacian, that is, recognizing focuses that are nearby extrema as for both space and scale, in the discrete case by examinations with the closest 26 neighbors in a discretized scale-space volume. The distinction of Gaussians administrator can be viewed as an estimation to the Laplacian, with the verifiable standardization in the pyramid likewise establishing a discrete guess of the scale-standardized Laplacian.

Keypoint Localzation

Scale-space extrema identification delivers such a large number of keypoint competitors, some of which are shaky. The subsequent stage in the calculation is to play out an elaborated fit to the adjacent information for exact area, scale, and proportion of central shapes. This data enables focuses to be rejected that have low difference (and are in this way delicate to clamor) or are ineffectively confined along an edge.

Interpolation of nearby data for accurate position

In the first place, for every competitor keypoint, insertion of adjacent information is utilized to precisely decide its position. The underlying methodology was to simply find each keypoint at the area and size of the competitor keypoint. The new methodology computes the introduced area of the extrema, which considerably enhances coordinating and steadiness. The insertion is finished utilizing the quadratic Taylor development of the Difference-of-Gaussian scale-space work, $D(x,y,k\sigma)$ with the origin. This Taylor expansion is given by[40]:

$$D(x) = D + \frac{\partial D^T}{\partial x}x + \frac{1}{2}x^T \frac{\partial^2 D}{\partial x^2}x \quad (3.3)$$

Where D and its subordinates are assessed at the competitor keypoint. Additionally, $x = (x, y, \sigma)^T$ is the counter balance. The area of the extreme $\hat{x}$ is controlled by taking the subordinate of this capacity concerning x and setting it to zero. On the off chance that the offset $\hat{x}$ is bigger than 0.5 in any measurement, at that point that means that the extremum lies nearer to another hopeful keypoint. For this situation, the competitor keypoint is changed and the insertion performed rather about that point. Generally the balance is added to its hopeful keypoint to get the interjected gauge for the area of the extrema. A comparable sub-pixel assurance of the areas of scale-space extrema is performed in the continuous usage dependent on half breed pyramids created by Lindeberg and his associates.

Eliminating edge responses

For ineffectively characterized tops in the DoG work, the foremost ebb and flow over the edge would be a lot bigger than the central ebb and flow along it. Finding these foremost bends adds up to understanding for the eigenvalues of the second-arrange Hessian matrix, $H[40]$:

$$H = \begin{pmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{pmatrix} \quad (3.4)$$

The eigenvalues of H are corresponding to the vital ebbs and flows of D . For reasons unknown, the proportion of the two eigenvalues, say α is the bigger one, and β the lesser one, with ratio $r = \alpha/\beta$, is sufficient for SIFT's cases. We get the sum of two eigenvalue from the trace of H , $(D_{xx} + D_{yy})$ when it is the determinant, such as $D_{xx}D_{yy} - D_{xy}^2$, yields the product.

The ratio $R = \text{Tr}(H^2/\text{Det}(H))$ can be equal to $(r + 1^2)/r$, which depends just on the proportion of the eigenvalues as opposed to their individual qualities. R is least when the eigenvalues are equivalent to one another. Along these lines, the higher the supreme distinction between the two eigenvalues, which is proportionate to a higher total contrast between the two important bends of D , the higher the estimation of R . It follows that, for some threshold eigenvalue r_{orth} , if R for a candidate keypoint is larger than $(r_{\text{th}} + 1)^2/r_{\text{th}}$, that keypoint is inadequately confined and consequently dismissed. The new approach uses $r_{\text{th}} = 10$.

Orientation assignment

In this progression, each keypoint is allocated at least one introductions dependent on neighborhood picture inclination bearings. This is the key advance in accomplishing invariance to pivot as the keypoint descriptor can be spoken to with respect to this introduction and along these lines accomplish invariance to picture revolution.

First, the Gaussian-smoothed image $L = (x, y, \sigma)$ at the keypoint's it takes σ , so all calculations are performed in a scale-invariant way. We can take a sample image $L(x, y)$ at scale σ , where the gradient magnitude, $m(x, y)[40]$, and orientation, $\theta(x, y)[40]$, are computed using pixel differences:

$$m(x, y) = \sqrt{(L(x+1, y) - L(x-1, y))^2 + (L(x, y+1) - L(x, y-1))^2} \quad (3.5)$$

$$\theta(x, y) = \text{atan2}(L(x, y+1) - L(x, y-1), L(x+1, y) - L(x-1, y)) \quad (3.6)$$

The greatness and direction estimations for the angle are improved the situation each pixel in a neighboring district around the keypoint in the Gaussian-obscured picture L . An

introduction histogram with 36 bins is shaped, with each canister covering 10 degrees. Each example in the neighboring window added to a histogram receptacle is weighted by its slope extent and by a Gaussian-weighted roundabout window with σ that is 1.5 occasions that of the size of the keypoint. The crests in this histogram relate to predominant introductions.

When the histogram is filled, the introductions relating to the most astounding pinnacle and neighborhood crests that are inside 80% of the most elevated pinnacles are doled out to the keypoint. On account of various introductions being allotted, an extra keypoint is made having indistinguishable area and scale from the first keypoint for each extra introduction.

Fig. 3.4 Feature extracted from multiple images: Arrows indicates features

After scale space extrema are detected, the SIFT algorithm discards low contrast keypoints and then filters out those located on edges. Resulting set of keypoints is shown in Figure 3.4.

Despite the fact that the component of the descriptor, such as 128, appears to be high, descriptors with lower measurement than this do not execute too over the scope of coordinating assignments and the computational cost stays low due to the estimated BBF (see beneath) technique utilized for finding the closest neighbor. Longer descriptors keep on improving the situation however not by much and there is an extra risk of expanded affectability to contortion and impediment. It is additionally demonstrated that features coordinating exactness is above half for perspective changes of up to 50 degrees. Hence, SIFT descriptors are invariant to minor relative changes.

3.1.6 Approximate Nearest Neighbors (ANN) Algorithm

Approximate nearest neighbor search algorithm is permitted to return focuses, whose separate from the inquiry is at most c times the separation from the question to its closest focuses. The intrigue of this methodology is that, as a rule, an inexact closest neighbor is nearly comparable to the correct one. Specifically, in the event that the separation measure precisely catches the thought of client quality, little contrasts out yonder ought not to make any difference.

Fig. 3.5 p_j is the nearest point to q . In the c -approx nearest neighbor problem any point within the radius cr is accepted.

From Figure 3.5 In the approximate nearest neighbor search problem, we relax our goal of exactly finding the closest point p_i . Instead we look for p_i , so that $d(q, p_i) \leq c \cdot \min_j d(q, p_j)$. In this section by using Locality Sensitive Hashing (LSH), we introduce schemes that can answer to approximate NNS in sub-linear query time.

Locality Sensitive Hashing

Kushilevitz et al. [10] show that one can answer $1 + \epsilon$ approximate nearest neighbor with poly-logarithmic query time and $\text{poly}(n)$ preprocessing time and storage.

Indyk and Motwani [11] consider the following problem: Can we answer the approximate

NNS with query time $o(n)$ with small amount of storage? To achieve this they use Locality Sensitive Hashing (LSH).

Definition 1- A family of hash functions is (r, cr, p_1, p_2) -LSH with $p_1 \geq p_2$ and $c > 1$ if,

- $P r [h(x) = h(y)] \geq p_1$ when $d(x,y) \leq r$ (close points).
- $P r [h(x) = h(y)] \leq p_2$ when $d(x,y) \geq cr$ (distant points).

An (r, ce, p_1, p_2) LSH can be used to design an algorithm for approximate NNS. For now instead of NNS we focus on the following problem: If there is some point within distance r of q , we want to report a point within distance cr (For the NNS we can store multiple copies of this scheme for different values of r).

Assume we have a family of hash functions with (r, ce, p_1, p_2) -LSH property. We sample k hash functions independently from the family and produce for each element in a single hash value[10]:

$$g(x) = h_1(x)h_2(x)....h_k(x) \quad (3.7)$$

This is a concatenation of the k hash values. We say that we have a collision between $x \neq y$ if $g(x) = g(y)$. Since, g is a concatenation of k hash functions this can happen only if $h_i(x) = h_i(y)$ for all $i \in [k]$. Now let us compute the probability of collision:

- If $d(x, y) \leq r$ then the probability of collision would be at least p_1^k .
- If $d(x, y) \geq cr$ then the probability of collision would be at least p_2^k .

We choose k so that $p_2^k = 1/n$. Then the probability that a bad point p , i. e. $d(q, p) > cr$, maps to the same bucket as query q is small. To achieve this one should set $k = \log(n)/\log(1/p_2)$.

Now let us compute the collision probability for a close point. We can write $p_1 = p^p$ for some $p < 1$, then[10]:

$$p_2^k = 1/n \Rightarrow p_1^k = 1/n^p \quad (3.8)$$

Which is quite small. To correct this we produce multiple hash tables, namely $L \approx n^p$ hash tables, so that with high probability we get some close point in at least one of the hash tables. To summarize, this method uses:

- n^p hash tables,
- n^p query time,

- $n^{1+\rho}$ storage.

Here ρ is the best parameter that we can find over all sets of locality sensitive hashing schemes, and is therefore a function of the constant c and metric d , i.e. $\rho = f(c, d(\cdot, \cdot))$. And since usually ρ is not small, both storage and query time are not too small in this scheme.

In two papers by Panigrahy [12] and Lv et al. [13] the authors try to reduce the amount of storage. The underlying idea is the same in the sense that they try to look up more than one entry of a hash function for a query. In Panigrahy's paper [12], the algorithm adds random perturbations to the original query and looks up for all generated points in a single hash function. In Lv et al.'s [13] paper they introduce an indexing scheme called multi-probe which uses a probing sequence to look up multiple entries that have a high probability of containing the nearest neighbors of a query point. In these schemes the number of hash tables

is reduced to $\sim(1)$, and the query time is p' . In a paper by Kapralov [14], it is shown that $O(n$

ρ for Hamming Metric

In this section we compute ρ for hamming metric in $\{0, 1\}^d$. A simple hash function is to pick the value of a random coordinate. By this hash function we have:

- if $d(x, y) \leq r$: $\text{pr}[h(x) = h(y)] \geq 1 - r/d \approx e^{-r/d}$
- if $d(x, y) \geq cr$: $\text{pr}[h(x) = h(y)] \leq 1 - r/d \approx e^{-r/d}$

Therefore $p_1 = p_2 = 1/2^{1/c}$ and $\rho = 1/c$. For hamming metric this turns out to be the best possible

ρ for c -approx NNS. Using a similar strategy one can show the same result for l_1 norm. That is if $d(x, y) = \|x - y\|_1$, then $\rho = 1/c$. We also know for $d(x, y) = \|x - y\|_2$ (l_2 norm), $\rho = 1/c^2$.

3.1.7 Matching Features

Coordinating features is a straightforward calculation that endeavors to distinguish a component in whatever remains of accessible pictures. This is an important procedure because of the need of process the triangulation in a later advance. One approach to keep away from scan for all pixels in each picture is to utilize epipolar lines. Accepting we have a pinhole model of camera, when two cameras see a similar 3D scene from two particular view-foci, there are various geometric relations between the 3D focuses and their projections onto the 2D pictures. This prompts imperative between the picture focuses. RANSAC algorithm has been used to corresponding or matching features.

Fig. 3.6 Feature Extraction, Colored Arrows indicates features.

Random Sample Consensus (RANSAC)

We have navigated to RANSAC algorithm for parameter estimation as well as to get the corresponding feature output. Random sample consensus (RANSAC) is an iterative technique to assess parameters of a numerical model from an arrangement of known and certain information that contains outliers, when outliers are to be concurred no effect on the estimations of the assessments [15]. Random sample consensus (RANSAC) algorithm is a general parameter estimation approach intended to adapt to a huge extent of anomalies in the info information. Not at all like a considerable lot of the regular hearty estimation strategies, for example, M-estimators and slightest middle squares that have been received by the PC vision network from the insights writing. RANSAC is a resampling procedure that creates hopeful arrangements by utilizing the base number perceptions [16].

Information focuses required to assess the basic model parameters. Not at all like traditional testing systems that utilization however much of the information as could be expected to acquire an underlying arrangement and afterward continue to prune exceptions, RANSAC utilizes the littlest set conceivable and continues to augment this set with steady information focuses.

Here we can do N numbers of equations. P is the probability that ensures that at least one example does not include an outlier. Give u a chance to speak to the likelihood that any chosen information point is an inlier what's more, $v = 1 - u$ the likelihood of watching an anomaly. N cycles of the base number of focuses signified m are required, where [16]

$$1 - p = (1 - u^m)^N \quad (3.9)$$

after manipulating [16],

$$N = \frac{\log(1 - p)}{\log(1 - (1 - v^m))} \quad (3.10)$$

3.1.8 GUI Application

The graphical user interface (GUI) is a type of UI that enables clients to communicate with electronic gadgets through graphical symbols and visual markers, for example, optional documentation, rather than content based UIs, composed direction names or content route.

Generating 3D Point Cloud

We started off by downloading the visualSFM which is a GUI application for 3D reconstruction using structure from motion. We install the OSM Bundler files. After installing and opening the zip folder we copied the bin files to the folder that has the visualSFM.

After opening the visualSFM software, we clicked on the open multiple images button and selected clear images from our computer library. It's important for the pictures to be extremely clear since blurry images will not produce satisfactory result. Afterwards, we clicked on compute missing matches button. We monitored the process starting in the log window. We then clicked on compute 3D reconstruction. We were able to zoom in and out as well as flip the dotted figure and locate the multiple cameras with our cursor. We did bundle adjustment which was pretty fast. We clicked on run dense reconstruction next and it prompted us to name and save our file. We started analyzing our 3d model and cleaning up the edges of our model which is one of the very few things that can be done in VisualSFM. We can select parts of the model and delete them to give it a cleaner look.

We install the open source 3D point cloud and mesh processing software, Meshlab. After installing the software, we opened our previously saved visualSFM model file under MeshLab. It has precise pinpoint options for cropping the model furthermore. The software gave us 2 different models, one the actual part we wanted to keep and another with the cropped out leftover parts. The program does not have an undo option. So all changes made are final, therefore, it's a good practice to save the leftover model.

We explored a lot of different things with our model. We calculated the density, curvature and roughness of the points. We tried extracting segments and cross sections of the model. We were also able to rasterize our model and turn it into a picture. We could flash some light into the model or turn it off. Last but not least we tried comparing our visualSFM model with a laser scan of the same image. We selected both models and clicked on registration under tools. We selected match scale to align the two models. It's important to do so since our two comparable models should be roughly the same size. The next step was to do fine registration. We selected adjust scale and 100% as the final overlap since the overlap looked pretty close. After the program finished calculating the final RMS, it returned the metrics. We also measured cloud to cloud distance. After going through all those process we got dense 3D model successfully. We then import dense 3D mesh to Meshlab and performed texturing and got the final test data.

Fig. 3.7 Test data: From top left : (a) Snapshot from 0 degree angle, (b) Snapshot from 30 degree (left Rotation) angle, (c) Snapshot from 60 degree (left Rotation)angle, (d)Snapshot from 90 degree (left Rotation) angle From bottom left: (e) Snapshot from 30 degree(right Rotation) angle, (f) Snapshot from 60 degree(right Rotation) angle and (g) Snapshot from 90 degree(right Rotation) angle

As described in Fig 3.7 We took multiple snapshot from 0 degree, 30 degree (left rotation), 60 degree (left rotation), 90 degree (left rotation) and then again reversely in 30 degree (right rotation), 60 degree (right rotation), and 90 degree (right rotation). Through all these we calculated our test accuracy and saved all these shots as our Test Data.

3.2 Data Training

3.2.1 Dataset Creator Algorithm

To train our system we created our own dataset with Dataset Creator Algorithm. The Algorithm uses Haar Cascade Algorithm and Detect MultiScale Algorithm to create dataset. As explained in Figure 8, it takes an image and convert it into grayscale. Then, Haar cascade algorithm Detect Face from the grayscale image and Detect MultiScale Algorithm resizes the image and consider only the region which have the faces. For a single input RGB

Fig. 3.8 Dataset Creator Algorithm to create Grayscale Face DataSet.

image we get 20 images as output which were converted to Gray scale and only the face region were considered.

Fig. 3.9 Workflow of Dataset Creator Algorithm.

As in Figure 9 it takes an image and converts it into gray scale using OpenCVs' built-in `cvtColor` method. Then, it uses Haar Cascade Face Detector Algorithm [17] to detect faces and Detect MultiScale resizes the region of interest and creates multiple images of faces.

3.2.2 Training Dataset using HAAR Cascade Algorithm

Based on the concept of the feature extraction Paul Viola and Michael Jones proposed Haar Cascade, a machine learning Object Detection algorithm to identify objects in images and videos. A cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.

The algorithm has four steps:

1. Haar Feature Extraction
2. Integral Images
3. The local Binary Patterns
4. Adaboost Training
5. Cascading Classifiers

3.2.3 Haar Feature Selection

An adjacent rectangular regions at a particular image in a detection window is considered. Then, difference between sums are measured and sums come from the pixel intensities of each regions. Subsections of an image is categorized using this difference. Haar feature's principle is based on detection of features encoding of some information about the class to be detected. They are adjacent rectangles in a particular position of an image. Figure 3.10 shows the types of Haar features depending on the number of adjacent rectangles.

According to [18] there are three types of Haar features.

1. Edge feature (Two upper squares of Figure 3.10).
2. Line feature (Lower left square of Figure 3.10).
3. Center-surround feature (Lower right square of Figure 3.10).

As described in [19], the idea behind Haar-like feature selection algorithm is simple. It lies on the principle of computing the difference between the sum of white pixels and the sum of black pixels. The main advantage of this method is the fast sum computation using the integral image. It is called Haar-like because it is based on the same principle of Haar wavelets [18].

Fig. 3.10 Rectangular Feature.

3.2.4 Integral Image

An intermediate representation of the image called integral image [20] computes the rectangle features. It consists of small unit of the given image.

Fig. 3.11 Integral Image blocks

For example (see Figures 3.11 and 3.12), the value of this integral image at location 1 is the sum of pixels in rectangular W. The value at location 2 is $W+X$ and so on. So the sum of pixels in rectangular Z is [20]

$$\text{Sum}(Z) = \text{ii}(4) - \text{ii}(3) - \text{ii}(2) + \text{ii}(1) \quad (3.11)$$

where $\text{sum}(Z)$ is the sum of pixels in the rectangular Z only, which is the sum of pixels in the rectangle $W+X+Y+Z$, represented by $\text{ii}(4)$, minus $\text{ii}(3)$, which is the integral image of rectangle $W+Y$, minus $\text{ii}(2)$, which is the integral image of $W+X$, and finally we had $\text{ii}(1)$, which is the integral image of the rectangle W. The addition is performed because the region W is subtracted twice in $\text{ii}(3)$ and $\text{ii}(2)$.

The example in Figure 3.13 shows the integral transformation of images.

According to Figure 3.13, original images are represented by blue matrices and the other matrices represent integral images. The integral value of the darker part in the first matrix is $5+2+3+1+5+4 = 20$ and the integral value of darker part of third example is $5+4+2+2+1+3 = 17$. In the fourth matrix by subtracting the integral image up to pixel 3, minus the integral

Fig. 3.12 Integral Image Sum

Fig. 3.13 Integral Image: illustration example

image of the regions outside the shaded area starting from the upper left pixel. Therefore, the integral image of the shaded area is 34, which represents the shaded area up to pixel 3, minus 14, which is the integral image of the region above the shaded area, minus 8, which is the integral image of the region on the left of the shaded area, plus 5, which was included in both integral value subtractions (14 and 8), which yields $34 - 14 - 8 + 5 = 17$. The integral image is defined as follows[20]:

$$ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y') \quad (3.12)$$

Here, $ii(x, y)$ is the integral image and $i(x', y')$ is the original image.

Therefore, the integral value of a specific pixel is the sum of pixels on the top of it towards the left [20, 21]. Then the image can be integrated in fewer pixel operations, since the traversing starts on the top left towards the bottom right.

3.2.5 The Local Binary Patterns (LBP)

The Local Binary Patterns is a simple texture operator. It is a very efficient way which labels the pixels by thresholding neighborhood of a particular pixel and get the results in binary number. The most important property of the LBP in real-world applications is its robustness to monotonic gray-scale changes caused by illumination variations and others. Again, its computational simplicity is another property for what it is well known [21]. However, it has two major stages to discuss-

a) Local Binary Patterns Operator

According to [22] for each pixels eight neighboring pixels is considered (Figure 3.14). Then neighbor pixels are valued 0 or 1 based on the gray-scale value of the subject pixel. As a result each pixel have a string of binary values consisting eight bit. Figure 3.15 shows an example of the calculation.

Fig. 3.14 Example of primitives detected by LBP

Fig. 3.15 LBP calculation Example

Again, uniform Local Binary Pattern contains at most two bitwise transitions from 0 to 1 or vice versa when the bit pattern is traversed circularly. For example, the patterns 11111111 (0 transitions), 10001111 (2 transitions), and 00110000 (2 transitions) are uniform whereas the patterns 11001001 (4 transitions) and 10101101 (6 transitions) are not.

The following notation is used for the LBP operator: $LBPP_{Ru2}$. The subscript represents using the operator in a (P, R) neighborhood. Superscript u2 stands for using only uniform patterns and labeling all remaining patterns with a single label. After the LBP labeled image $f(x, y)$ has been obtained, the LBP histogram can be defined as[22]:

$$H_i = \sum_{x,y} I\{f(x, y) = i\}, i = 0, \dots, n-1 \quad (3.13)$$

In which n is the number of different labels produced by the LBP operator, and $I\{A\}$ is 1 if A is true and 0 if A is false. When the image patches whose histograms are to be compared have different sizes, the histograms must be normalized to get a coherent description[22]-

$$N_i = \frac{H_i}{\sum_{j=0}^{n-1} H_j} \quad (3.14)$$

b) LBP Detection Algorithm

The calculation utilized is a variation of the course calculation presented by Viola and Jones [20] and utilizes LBP rather than Haar-features with the end goal to have quicker handling and supported classifiers. The LBP algorithm slides its handling window over the object picture for assessing the progressive phases of the cascade algorithm by scoring their highlights. Each features is depicted by 3×3 neighboring rectangular regions.

The estimation of each feature is processed by contrasting the focal region and the neighboring region around it (8 neighbors). The outcome is as an 8-bit paired esteem called LBP. Various features speak to a phase of the cascade algorithm. Each feature has positive and negative weights related with it. For the situation where the feature is in consistence with

the object to be identified, the positive weight is added to the total. For the situation where the feature is conflicting with the object, the negative esteem is added to the aggregate. The entirety is then contrasted with the edge of the stage. In the event that the total is underneath the limit, the stage fails and the cascade ends early, and, in this manner, the handling window moves to the following window. In the event that the total is over the limit, the following phase of the cascade is endeavored. All in all, if no stage rejects a competitor window, it is accepted that the question has been identified.

In order to avoid the redundancy of computing the integral of rectangles, the integral images are calculated to speed up the calculation of the feature.

3.2.6 AdaBoost Learning Algorithm

Viola and Jones [20] used AdaBoost learning algorithm to select a specific Haar-like feature as a threshold. AdaBoost is used to create strong classifier from combining a collection of weak classification functions [23]. The strongest classifier uses the strongest feature, which is the best Haar-like feature, that is, the feature that best separates the positive and negative samples. Thinking about a mono-arrange classifier and given an arrangement of features, we can manufacture a face detector by applying every one of the covers at each picture area (each move and each scale). For this a wide range of learning techniques could be utilized.

Additionally, we have a total set of 36550 features which is far bigger than the quantity of pixels, so regardless of whether the features reactions are extremely easy to figure (prominently with the vital picture portrayal), applying the all sets of features would be two times costly in time. The following stage in the working of the face detector is therefore to utilize a learning capacity which chooses a little arrangement of these features: the ones which separated the best positive and negative precedents. The subsequent final classifier would be a straightforward direct blend of these few Haar-like features.

For this, we will discuss in this segment around an algorithm called AdaBoost (Adaptive Boosting) which has two principle objectives:

1. Selecting a couple of set of features which speaks to and additionally conceivable appearances.
2. Train a solid final classifier with a linear combination of these best features. In the accompanying subsections, it is clarified why we have picked this algorithm rather than more established ones and afterward we some hypothesis is disclosed to demonstrate why AdaBoost is productive and how it tends to be adjusted to face detection.

Possible algorithms

Given a set of features and a training set of positive and negative models, any machine learning approach could be utilized to take in a characterization work. Here is a summary of the main conceivable methodologies which could be utilized to prepare a classifier:

- Mixture of Gaussian model
- Simple image features and neural network
- Support Vector Machine
- Winnow learning procedure.

AdaBoost is a productive boosting algorithm which combine basic measurable learners while decreasing fundamentally the training blunder as well as the more subtle speculation mistake. As all the learning functions, it presents advantages and drawbacks which are exposed here:

Advantages:

1. No a prior knowledge

AdaBoost is a calculation which just needs two inputs of info: a training dataset and an arrangement of features (order capacities). There is no need any from the earlier learning about face structure. The most agent feature will consequently be chosen amid the learning

2. Adaptive algorithm

At each phase of the taking in, the positive and negative precedents are tried by the present classifier. In the event that a model x_i is misclassified, that implies that it is difficult to group. It cannot clearly be assign in the great class. With the end goal to build the discriminant intensity of the classifier these misclassified precedents are up-weighted for the following calculation cycles.

So the effectively arranged precedents are recognized in the principal cycles and will have less weight in the learning of the following stages to concentrate on the harder models.

3. The training error theoretically converge exponentially towards 0.

As demonstrated by Freund and Schapire in [24], given a limited arrangement of positive and negative models, the preparation mistake achieves 0 out of a limited number of emphases.

Drawbacks:

1. The result depends on the data and weak classifiers

The nature of the final location depends exceptionally on the consistence of the preparation set. Both the measure of the sets and the interclass fluctuation are essential components to consider. Other way, the sorts of fundamental classifiers which are joined have some influence on the outcome.

The main requirement for all the fundamental capacities is to be superior to arbitrary determination however in the event that we need to accomplish great location rates in a fitting number of emphases, they must be too picked as could be expected under the circumstances.

2. Quite slow training

At every cycle step, the calculation tests every one of the features on every one of the models which requires a calculation time specifically relative to the span of the features and precedents sets. Envision that the preparation set has a large number of positive and negative precedents and a total arrangement of 36550 features. Be that as it may, the calculation time is expanded directly with the measure of the two sets.

History of Boosting and AdaBoost methods

The Boosting hypothesis takes its foundations in the PAC learning [25]. They demonstrated that a mix of straightforward learners, just superior to anything arbitrary could yield a decent last theory. That is the primary thought of what is called Boosting. AdaBoost (Adaptive Boosting) was presented as a practical algorithm of the Boosting hypothesis.

Let $h_1, h_2, \dots, h_r$ be a set of simple hypothesis, considers the composite ensemble of hypothesis [25]:

$$f(x) = \sum_{t=1}^T a_t h_t(x) \quad (3.15)$$

a_t denotes the coefficient with which the ensemble member h_t is combined. Both a_t and h_t have to be learned during the boosting process. In the first place, Boosting calculations were acknowledged for their exhibitions with low clamor information. Be that as it may, the principal calculations furnished too terrible outcomes with uproarious examples because of over-fitting so the utilizations of Boosting were restricted.

AdaBoost asymptotically accomplishes an extensive edge grouping, that implies that it focus its assets on a couple of difficult to-learn designs that are strikingly fundamentally the same as help vectors [26]. Endeavoring to enhance the strength of Boosting, it was

fascinating to clear up the relations between Optimization theory and Boosting methods. From here, it ended up conceivable to characterize Boosting algorithms for relapses [27], multi class issues, unsupervised learning and to set up combination proofs for boosting calculations by utilizing results from the Theory of Optimization.

Algorithm 1 The AdaBoost Algorithm [28]

Input: $S = \{(x_1, y_1), \dots, (x_n, y_n)\}$ Number of iterations T .

1: Initialize: $d(1) = 1/N$ for all $n = 1, \dots, N$

2: Train classifier with respect to the weighted sample set $\{S, d^{(t)}\}$ and obtain hypothesis:

$h_t : x \rightarrow \{-1, +1\}$, i.e. $h_t = L(S, d^{(t)})$

3: Calculate the weighted training error ϵ_t of h_t :

$$\epsilon_t = \sum_{n=1}^N d_n^{(t)} I(y_n \neq h_t(x_n))$$

4: Set $a_t = \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t}$

5: Update the weights: $d_n^{(t+1)} = d_n^{(t)} \exp\{-a_t y_n h_t(x_n)\} / Z_t$
where Z_t is a normalization constant, such that, $\sum_{n=1}^N d_n^{(t+1)} = 1$

6: Break if $\epsilon_t = 0$ or $\epsilon_t \geq 1/2$ and set $T = t - 1$.

Output: $f_T(x) = \sum_{t=1}^T \frac{a_t}{\sum_{t=1}^T a_t} h_t(x)$

Adaptation to face detection

This new subsection will explain how the algorithm can be adapted to our face detection context, particularly with the introduction of an asymmetric classification.

AdaBoost, is a an algorithm which minimize the generalization error but it does not minimize the number of false negative .There are several methods to modify AdaBoost in order to obtain an asymmetric algorithm, asymmetric in the sense that we want to increase the influence of the positive examples which have been misclassified earlier in the process in order to minimize the false negative rate which are missed.

One first simple mean is unbalance the underlying dissemination of the positive and negative models as in [25]. In the event that we need to limit the false negatives, we can build the weight on positive models with the goal that the base mistake criteria will likewise have low false negatives.

This idea can be introduced by changing the loss function in a non- symmetric loss function. Recall that the classical AdaBoost minimizes[27]

$$\prod_t Z_t = \sum_i \exp(-y_i \sum_t h_t(x_i)) \quad (3.16)$$

Each term in the summation is bounded above by the loss function[27]

$$\exp(-y_i \sum_t h_t(x_i)) \geq \lambda_i(y_i, f(x_i)) = I[y_i \neq f(x_i)] \quad (3.17)$$

Where, λ is the loss function. It follows that minimizing $\prod_t Z_t$ minimizes an upper bound on simple loss. So we can introduce the asymmetric loss defined by[27]:

$$A\lambda_i = \begin{cases} k & \text{if } y_i = 1 \& f(x_i) = -1 \\ \frac{1}{k} & \text{if } y_i = -1 \& f(x_i) = 1 \\ 0 & \text{otherwise} \end{cases} \quad (3.18)$$

Where false negative cost k times more than false positives. Note that, $A\lambda_i = \exp(y_i \log k) \lambda_i$. We can easily obtain the bounds of the asymmetric loss and we see[27]-

$$\exp(-y_i \sum_t h_t(x_i)) \cdot \exp(y_i \log \frac{1}{k}) \geq A\lambda_i \quad (3.19)$$

In order to minimize this bound, we can use a non-uniform weight initialization[27]:

$$d_1 = \exp(y_i \log(k)) / N \quad (3.20)$$

Updating the weights will become[27]:

$$D_{t+1}(i) = \frac{\exp(-y_i \sum_t h_t(x_i)) \cdot \exp(y_i \log(\frac{1}{k}))}{\prod_t Z_t} \quad (3.21)$$

The change of the pre-weighting is transmitted during that time term of the numerator. However, the effects of the unbalanced weights are lost after the first iteration. In fact, the AdaBoost algorithm seems to be too greedy. The first classifier absorbs the effects of the asymmetric weights.

AdaBoost chooses in this way a little set of features, great outcomes can be acquired with nearly 200 features. The main features chose in the process can be effortlessly translated: they stress on specific features of faces. The eye area is often darken than the front and the nose area is brighter than the eyes.

Notwithstanding, this is not sufficient to achieve the fixed objective of our project. The calculation time for a 200 features classifier is too large to fulfill goal. We will introduce a path with join classifiers in cascade with the end goal to concentrate rapidly on the locales of interest.

3.2.7 Cascade Classifier

We realize how to choose few basic features and to consolidate them into a solid classifier. Be that as it may, we have to present another principle commitment in our face identification framework with the end goal to decrease altogether the calculation time. This commitment is an attentional cascade which can besides accomplish better identification exhibitions. We have seen that it is conceivable to limit the quantity of false negatives rather than traditional preparing error that is definitely the fundamental thought that will be utilized to assemble this cascade classifier. The guideline is to dismiss rapidly the dominant part of negative windows while keeping every positive examples and after that emphasis on sub windows with more entire classifiers. To do that, the main stages in the cascade will contain just couple of features, which accomplish high location rates however will have a false positive rate of approximately 40 %. Unmistakably it is not adequate for a face discovery assignment yet joining huge numbers of these phases which are increasingly discriminant will allow to achieve the objective of quick face location.

We can simply contrast this cascade structure and a declined choice tree. On the off chance that a sub-window is named positive at one phase, it continues down in the cascade and will be assessed by the following stage. It will resemble this until this sub-window is discovered negative by one phase or if every one of the stages arrange it as positive. In this last case, it will at last be considered as a positive model.

Fig. 3.16 Cascade Classifier

Figure 3.16 - Schematic description of cascade detection. A series of classifiers is applied to every sub-window. The initial classifier eliminates a large number of negative examples with very little processing. Subsequent layers eliminate additional negatives but require additional computation. After several stages of processing, the number of sub-windows have been reduced radically. The examples classified as positive by the last stage are definitively considered as positive.

As clarified in a setting of the face discovery, the objective of the undertaking is to

recognize faces in pictures which contain few appearances. Seeing that there are about 20.000.000 windows in a 100×100 picture for just a couple of appearances, the incredible lion's share of windows are negative ones. So it is a genuine gain of time to decrease rapidly this number. Regardless of whether the last phases of the course depend on numerous thousand features, they will be called just for few sub-windows.

Training a cascade of classifiers

The objective of the cascade location is to accomplish given both false positive rate and discovery rate. The decision of these objectives is subjective. Ordinarily, past frameworks have accomplished location rates somewhere in the range of 85 and 95 percent and false positive rate on the request 10^{-5} . The quantity of features in each stage and the aggregate number of stages will depend of these limitations.

Let F be the false positive rate of the cascaded classifiers, K the number of classifiers and f_i the false positive rate of the i -th classifier on examples that get through to it. For a given trained cascade of classifiers, F is given by[25]:

$$F = \prod_{i=1}^k f_i \quad (3.22)$$

Then the detection rate can be computed as[25]:

$$D = \prod_{i=1}^k d_i \quad (3.23)$$

d_i is the detection rate of the i -th classifier on the examples that get through to it.

To fix the ideas, an example is given here. If we want to achieve a detection rate of 90 percent, we can build a 10 stage classifier in which each stage has a detection rate of 0.99.

Indeed, $0.9 \sim 0.99^{10}$. If each of these stage rejects 70 percent of negatives, the total false
 =
 positive rate is $0.3010 \sim 6.10^{-6}$.

The quantity of features assessed when examining genuine pictures is important a proba-bilistic procedure. Any window will advance down through the course, one classifier at any given moment, until the point when it is chosen that the window is negative or, in extremely uncommon cases, the window prevails in each test and is named positive. The conduct of this procedure is dictated by the circulation of the pictures of the test set. The fundamental instrument which can gauge the execution of a given classifier is its positive rate, which is the extent of windows which are named as conceivably containing the question of intrigue. Given various stages in the course K , the positive rate p_i of the i -th classifier.

Give n_i a chance to be the quantity of highlights in the i -th arrange. The normal number of highlights which are assessed is given by[25]:

$$N = n_0 + \sum_{i=1}^k (n_i \prod_{j<i} p_j) \quad (3.24)$$

We can see just couple of precedents are objects, that is the reason the positive rate is relatively equivalent to the false positive rate. The first AdaBoost calculation must be alter to guarantee the minimization of the false negative rate rather than the preparation blunder. One basic approach to force that is to alter the last edge. Expanding this edge will influence gravely the discovery rate and enhance the false positive rate, while the contrary will yield bring down recognition rate with higher false positive rate. The principle issue is that is has never been demonstrated that changing the AdaBoost hypothetical preparing limit save the assurances in term of speculation mistake.

The cascade structure has three main parameters that we have to determine:

- The total number of classifiers : K
- The number of features n_i of each stage i ,
- The threshold Θ_i of each stage i ,

Finding the three ideal parameters is very muddled in the event that we remember that we need to limit the calculation time of the aggregate characterization. The standard is to build the quantity of highlights and of stages until the point when the given identification objective are come to. Given the minimum acceptable rates f_i and d_i , the detection rate d_i is reached by decreasing the AdaBoost threshold Θ_i and this also directly affects f_i . We increase the number of features n_i in the i -th stage until f_i is obtained.

One central point for the productivity of the course learning is the administration of the sets amid the preparation. Normal preparing sets are utilized for the principal stage, and after that, at every cycle, the current stage classifier is assessed on an approval set with the end goal to limit sound false positive and false negative rates. It would obviously unskilled to evaluate F_i and D_i on the training set which was used to obtain the model because these values would be evaluated better. Then, at each stage we reinitialize the negative training set. Once the objectives F_i and D_i are reached for the stage i , the current model is tested on a large set chosen randomly and many false positive alarms are introduced into the negative training set for the stage $i - 1$.

As the new negative training set is made with examples which have been misclassified

Algorithm 2 Learning in cascade[25]

Input: Denition of the targets of the learning

- f the maximum acceptable false positive rate per stage.
- d The minimum acceptable detection rate per layer
- F_{target} The false positive rate desired at the end of the process.
- P set of positive examples
- N set of negative examples

1: Initialization:

- $F_0 = D_0 = 1$
- $i = 0$ the number of the current layer.

2: Main loop: while $F_i >$

$F_{\text{target}} \Rightarrow i \leftarrow i + 1$

$\Rightarrow n = n_0$

$\Rightarrow F_i = F(i - 1)$

$\Rightarrow$ While $F_i > f \times F_i - 1$

- $n_i \leftarrow n_i + 1$
- Train a classifier using AdaBoost with P and N as training set.
- Compute f_i and d_i for the current classifier with the validation set.
- Decrease the threshold of the classifier until the detection rate for i -th classifier is at least $d \times d_i - 1 : D_i \geq D_i - 1 \times d$

$\Rightarrow$ Empty the negative training set.

3: If $F_i > F_{\text{target}}$, evaluate the current cascade classifier on a set of negative examples and put any false detections into the set N .

by the stage i , the stage $i + 1$ will be built with examples which can be considered as hard examples.

3.3 Face Recognition

People perform face recognition consequently and for all intents and purposes with no exertion. In spite of the fact that it sounds like an extremely simple task for us, it has turned out to be an unpredictable task for a PC, as it has numerous factors that can weaken the precision of the techniques, for instance: brightening variety, low goals and impediment amongst other.

In computer science, face recognition is fundamentally the undertaking of recognizing

a person dependent on its facial picture. It has turned out to be extremely well known over the most recent two decades, basically due to the new strategies created and the high caliber of the present recordings/cameras. The face acknowledgment frameworks can work fundamentally in two modes:

- Verification or authentication of a facial image: it basically compares the input facial image with the facial image related to the user which is requiring the authentication. It is basically a 1x1 comparison.
- Identification or facial recognition: it basically compares the input facial image with all facial images from a dataset with the aim to find the user that matches that face. It is basically a 1xN comparison.

There are different types of face recognition algorithms, for example:

- Eigenfaces (1991)
- Local Binary Patterns Histograms (LBPH) (1996)
- Fisherfaces (1997)
- Scale Invariant Feature Transform (SIFT) (1999)
- Speed Up Robust Features (SURF) (2006)

Every technique has an alternate way to deal with concentrate the picture data and play out the coordinating with the info picture. Be that as it may, the strategies Eigenfaces and Fisherfaces have a comparable methodology and in addition the SIFT and SURF techniques.

For our exploration we utilized the most famous face acknowledgment Algorithm and that is Local Binary Patterns Histograms (LBPH). It was first depicted in 1994 (LBP) and has since been observed to be a ground-breaking feature for surface order. It has additionally been resolved that when LBP is joined with histograms of situated angles (HOG) descriptor, it enhances the identification execution significantly on some datasets. Using the LBP joined with histograms we can speak to the face pictures with a basic information vector. As LBP is a visual descriptor it can likewise be utilized for face acknowledgment undertakings, as can be found in the accompanying well-ordered clarification[21]. Steps of the algorithm:

1. Parameters:

the LBPH uses 4 parameters:

- Radius: the range is utilized to assemble the roundabout nearby paired example and speaks to the span around the focal pixel. It is normally set to 1.
- Neighbors: the number of sample points to build the circular local binary pattern. Keep in mind: the more sample points you include, the higher the computational cost. It is usually set to 8.
- Grid X: the number of cells in the horizontal direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.
- Grid Y: the number of cells in the vertical direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.

2. Training the Algorithm:

First, we need to train the algorithm. To do so, we need to use a dataset with the facial images of the people we want to recognize. We need to also set an ID for each image, so the algorithm will use this information to recognize an input image and give you an output. Images of the same person must have the same ID. With the training set already constructed, let's see the LBPH computational steps.

3. Applying the LBP operation:

The first computational step of the LBPH is to create an intermediate image that describes the original image in a better way, by highlighting the facial characteristics[21]. To do so, the algorithm uses a concept of a sliding window, based on the parameters radius and neighbors.

Fig. 3.17 workflow of LBP operator

Based on the Figure 3.17, it breaks it into several small steps and they are-

- A facial image in grayscale.

- A part of this image as a window of 3x3 pixels. It can also be represented as a 3x3 matrix containing the intensity of each pixel (0-255).
- Then, the central value of the matrix to be used as the threshold.
- This value will be used to define the new values from the 8 neighbors.
- For each neighbor of the central value (threshold), a new binary value is set. We set 1 for values equal or higher than the threshold and 0 for values lower than the threshold.
- Now, the matrix will contain only binary values (ignoring the central value). We need to concatenate each binary value from each position from the matrix line by line into a new binary value (e.g. 10001101).
- Then, we convert this binary value to a decimal value and set it to the central value of the matrix, which is actually a pixel from the original image.
- At the end of this procedure (LBP procedure), we have a new image which represents better the characteristics of the original image.

It can be done by using bilinear interpolation. If some data point is between the pixels, it uses the values from the 4 nearest pixels (2x2) to estimate the value of the new data point.

Fig. 3.18 Selected neighbors point and radius from the central value. a) a) contain 8 neighbors and radius is 2, b) contain 8 neighbors and radius is 1, c) contain 12 neighbors and radius is 2, d) contain 12 neighbors and radius is 3

4. Extracting the Histograms:

Using the image generated in the last step, we can use the Grid X and Grid Y parameters to divide the image into multiple grids, as can be seen in the following image: We can extract the histogram of each region as follows:

- As we have an image in grayscale, each histogram (from each grid) will contain only 256 positions (0 255) representing the occurrences of each pixel intensity.
- Then, we need to concatenate each histogram to create a new and bigger histogram. Supposing we have 8x8 grids, we will have $8 \times 8 \times 256 = 16.384$ positions in the final histogram. The final histogram represents the characteristics of the image original image.

5. Performing the face recognition:

In this step, the algorithm is already trained. Each histogram created is used to represent each image from the training dataset. So, given an input image, we perform the steps again for this new image and creates a histogram which represents the image[21].

- So to find the image that matches the input image we just need to compare two histograms and return the image with the closest histogram.
- We can use various approaches to compare the histograms (calculate the distance between two histograms), for example: Euclidean distance, chi-square, absolute value, etc. In this example, we can use the Euclidean distance (which is quite known) based on the following formula[21]:

$$D = \sqrt{\sum_{i=1}^n (\text{hist1}_i - \text{hist2}_i)^2} \quad (3.25)$$

- So the algorithm output is the ID from the image with the closest histogram. The algorithm should also return the calculated distance, which can be used as a confidence measurement.
- We can then use a threshold and the confidence to automatically estimate if the algorithm has correctly recognized the image. We can assume that the algorithm has successfully recognized if the confidence is lower than the threshold defined.

Chapter 4

Result Analysis

As mentioned before we worked on our own dataset. We took multiple images from multiple angle of a person and import those images in VisualSFM where we ran SFM algorithm. Throughout the process, image feature points are detected by the SIFT keypoint detector firstly. Second, the keypoint descriptors between the image pair are matched by an approximate nearest neighbors (ANN) algorithm. The fundamental matrix of the image pair is then robustly estimated by a random sample consensus (RANSAC) algorithm whereas the outlier matches are removed. After we estimate the fundamental matrix and find geometrically consistent matches for each pair, the matches are organized into tracks. Each track contains a connected set of matching keypoints across multiple images. Finally, a set of camera parameters and 3D structures are jointly recovered by a bundle adjustment. Thus we get 3D model of face as .ply file. We could not read the .ply file and took snapshot with 30 degree interval starting from 0 degree to 90 degree left and 90 degree right. We took images of 0, 30, 60, 90, -30, -60, -90 degree angular deviation and experiment our results.

However, we used dataset Creator Algorithm to create dataset to train our model. Then, we used AdaBoost learning algorithm which train our model. Lastly we used LBPH face recognition algorithm to recognize faces.

4.1 Comparative Study and Accuracy

We studied a comparative study on face detection methods[29]. According to [29], The main information about the experimental results of few of the discussed 3D and 2D methods are shown in table 1. Previous works shows a great comparative study in the field of detecting faces from 3D and 2D images. On the contrary, our works (shown in table 4.1) give an accuracy rate of 80-100% depending on the angular deviation of 3D face model.

Table 4.1 Experimental results of few of the discussed 3D and 2D methods

Authors	Type	Image / model size	No. of Sample	Accuracy rate
Blanz and Vetter [30]	2D-based	4,488 (PIE))+1,940 (FERET)	-	95 and 95.9
Lu et al. [31]	2D-based	64×64	220	85
Ansari & Motaleb [32]	Set of points	-	52	96.2
Cook et al. [33]	GMM(ICP)	4000 points	360(3D_RMA)	97.33
Irfanoglu et al.[34]	Set of points	4000 points	90(3D_RMA)	96.66
Medioni & woupotitsch [35]	ICP	-	700	98
Lu et al. [36]	ICP	320×240 (18,000 points)	113	95.5
Wang et al. [37]	Sphere-spin	200×200	31	91.68
Chang et al.[38]	Multimodal	640 × 480	-	98.8
Papatheodorou & Rueckert [39]	Multimodal	10,000	900	66–100

Table 4.2 Experiments & Result Analysis

Sample	0 Degree	30 De- gree Angular Devi- ation (left)	60 De- gree Angular Devi- ation (left)	90 De- gree Angular Devi- ation (left)	30 De- gree Angular Devi- ation (Right)	60 De- gree Angular Devi- ation (right)	90 De- gree Angular Devi- ation (right)
Sample 1
Sample 2
Sample 3
Sample 4
Sample 5
Sample 6
Sample 7
Sample 8
Sample 9
Sample 10

Table 4.2 shows the result of the research which represents the rate of accuracy of the system. It shows that, the system can successfully recognize faces from multiple angular deviation.

We determined Accuracy with the following formula:

$$\text{Accuracy} = \frac{\text{number of datapoints that successfully recognize faces}}{\text{sample space}} \times 100\% \quad (4.1)$$

Using equation 26 we calculated accuracy. Table 2 shows our samples give-

1. 100% accuracy for the image taken from 0 degree deviation.
2. 100% accuracy for the image taken from 30 degree (right) deviation.
3. 80% accuracy for the image taken from 60 degree (right) deviation (8 out of 10).
4. 0% accuracy for the image taken from 90 degree (right) deviation.
5. 100% accuracy for the image taken from 30 degree (left) deviation.
6. 90% accuracy for the image taken from 60 degree (left) deviation (9 out of 10).
7. 0% accuracy for the image taken from 90 (left) degree deviation.

As our result shows, our method can successfully detect and recognize faces up to 60 degree left and 60 degree right angular deviation, in total 120 degree angular deviation. Nevertheless, we tried to train our model to recognize face up to 75 degree angular deviation, but it did not work properly.

Here, table 4.1 shows the previous comparative study of others and table 4.2 represents our works. It is to be said our work has similarities with those previous works, but we did it with a different approach than them. Our work focuses on face recognizing from multiple angle when most of the previous works focused on detecting faces from 0 degree angular deviation. On the contrary our work can detect faces up to 120 degree angular deviation depending on 60 degree left and right deviation respectively. That makes our research different from them. Nevertheless, we used LBPH algorithm which is easy to implement in almost every environment where others uses heavy tools to do their work.

Chapter 5

Conclusion

In this thesis, we firstly constructed a 3D facial image from multiple 2D images. We then train the sample images and figured out the matches between a pair of 2D and 3D image and recognize the face from 3D image. Finally we tried to calculate the accuracy of our task from the matches of each pair image from different angular deviation. The system that has been offered in this paper will give a good solution to the cases of face recognition in 3D images. In this whole process the most challenging part that we faced was finding out matches between 2D and 3D images. Here we were not completely satisfied with our job. We used LBPH algorithm for face recognition in 3D images. Again for data training we used Ada Boost which did not give us total satisfactory result. In these two cases we have planned to use Convolutional Neural Network (CNN) algorithm which we are expecting will make our whole research better and stronger.

5.1 Difficulties

We faced different difficulties through our research. Firstly, the 3d models we generate were not textured well. We used Meshlab to texture, render and filter our model to get better 3d mesh. Secondly, we could not manage to read our 3d mesh directly in our system. So, as our supervisor advised we took snapshot from multiple angular deviation to testify our research results.

5.2 Future Plan

We used AdaBoost learning algorithm and Haar cascade classifier as they are way faster and easier to implement. However we think there are some other algorithms like CNN,

RCNN and so on which can give us better result. If we could use CNN algorithm there was a probability to get more accurate result in cases of training data and recognizing them from 3D models. So, in future we will try to try with other machine learning algorithm.

References

- [1] Guan, Y. (2007, April). Automatic 3d face reconstruction based on single 2d image. In *Multimedia and Ubiquitous Engineering, 2007. MUE'07. International Conference on* (pp. 1216-1219). IEEE.
- [2] Medioni, G., & Fidaleo, D. (2012). "3d face reconstruction from 2d images". U.S. Patent No. 8,126,261. Washington, DC: U.S. Patent and Trademark Office.
- [3] Fidaleo, D., & Medioni, G. (2007, October). Model-assisted 3d face reconstruction from video. In *International Workshop on Analysis and Modeling of Faces and Gestures* (pp. 124-138). Springer, Berlin, Heidelberg.
- [4] Jiang, D., Hu, Y., Yan, S., Zhang, L., Zhang, H., & Gao, W. (2005). Efficient 3D reconstruction for face recognition. *Pattern Recognition*, 38(6), 787-798.
- [5] Reiter, M., Donner, R., Langs, G., Bischof, H. (2006, August). 3D and infrared face reconstruction from RGB data using canonical correlation analysis. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on* (Vol. 1, pp. 425-428). IEEE.
- [6] Lee, P. H., Huang, J. W., & Lin, H. Y. (2012, November). 3D model reconstruction based on multiple view image capture. In *Intelligent Signal Processing and Communications Systems (ISPACS), 2012 International Symposium on* (pp. 58-63). IEEE.
- [7] Shalaby, A., Elmogy, M., & El-Fetouh, A. A. (2017). Algorithms and Applications of Structure from Motion (SFM): A Survey. *Algorithms*, 6(06).
- [8] Lasang, P., Shen, S. M., & Kumwilaisak, W. (2014, September). Combining high resolution color and depth images for dense 3D reconstruction. In *Consumer Electronics-Berlin (ICCE-Berlin), 2014 IEEE Fourth International Conference on* (pp. 331-334). IEEE.
- [9] Torres, J. C., Arroyo, G., Romo, C., De Haro, J. (2012, September). 3D Digitization using structure from motion, *CEIG-Spanish Computer Graphics Conference*.
- [10] Kushilevitz, Eyal, Rafail Ostrovsky, & Yuval Rabani. "Efficient search for approximate nearest neighbor in high dimensional spaces." *Proceedings of the thirtieth annual ACM symposium on Theory of computing*. ACM, 1998.
- [11] Indyk, Piotr, & Rajeev Motwani. "Approximate nearest neighbors: towards removing the curse of dimensionality." *Proceedings of the thirtieth annual ACM symposium on Theory of computing*. ACM, 1998.

- [12] Panigrahy, Rina. "Entropy based nearest neighbor search in high dimensions." Proceedings of the seventeenth annual ACM-SIAM symposium on discrete algorithm. Society for Industrial and Applied Mathematics, 2006.
- [13] Lv, Qin, et al. "Multi-probe LSH: efficient indexing for high-dimensional similarity search." Proceedings of the 33rd international conference on Very large data bases. VLDB Endowment, 2007.
- [14] Kapralov, Michael. "Smooth tradeoffs between insert and query complexity in nearest neighbor search." Proceedings of the 34th ACM Symposium on Principles of Database Systems. ACM, 2015.
- [15] Broder, Andrei Z. "On the resemblance and containment of documents." Compression and Complexity of Sequences 1997. Proceedings. IEEE, 1997.
- [16] Derpanis, K. G. (2010). Overview of the RANSAC Algorithm. Image Rochester NY, 4(1), 2-3.
- [17] P. Viola & M. J. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features" Accepted Conference on Computer Vision and Pattern Recognition 2001
- [18] R. Leinhardt & J. Maydt, "An extended set of haar-like features for rapid object detection," in Proceedings of the International Conference on Image Processing (ICIP '02), vol. 1, pp. I-900–I-903, IEEE, Rochester, NY, USA, September 2002.
- [19] S. Guennouni, A. Ahaitouf, & A. Mansouri, "Multiple object detection using OpenCV on an embedded platform," in Proceedings of the 3rd IEEE International Colloquium in Information Science and Technology (CIST '14), pp. 374–377, IEEE, Tetouan, Morocco, October 2014.
- [20] P. Viola & M. J. Jones, "Robust real-time face detection," International Journal of Computer Vision, vol. 57, no. 2, pp. 137–154, 2004.
- [21] R. Benenson, M. Mathias, R. Timofte, & L. Van Gool, "Pedestrian detection at 100 frames per second," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '12), pp. 2903–2910, Providence, RI, USA, June 2012.
- [22] T. Ojala, M. Pietikäinen, & T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 7, pp. 971–987, 2002.
- [23] M.-T. Pham & T.-J. Cham, "Fast training and selection of Haar features using statistics in boosting-based face detection," in Proceedings of the 11th International Conference on Computer Vision (ICCV '07), pp. 1–7, IEEE, Rio de Janeiro, Brazil, October 2007.
- [24] Yoav Freund & Robert and E. Schapine. A decision-theoretic generalization of on-line learning and an application to boosting. In Computational Learning Theory: Eurocolt '95, pages 23-37. Springer-Verlag, 1995
- [25] L.G Valiant. A theory of the learnable. Communications of the ACM, 27(11):1134-1142, November 1984.

- [26] G. Rätsch, T. Onoda K-R. Müller. Soft Margins for AdaBoost, Machine Learning, 1-35, August 1998.
- [27] Duy & D.P Helmbold. Boosting methods for regression. Technical report, Department of Computer Science, University of Santa Cruz, 2000
- [28] Y. Freund & R.E Schapire. Game theory, on-line prediction and boosting In Proc. COLT, pages 325-332, New York, NY, 1996. ACM Press.
- [29] Abate, A. F., Nappi, M., Riccio, D., & Sabatino, G. (2007). 2D and 3D face recognition: A survey. Pattern recognition letters, 28(14), 1885-1906.
- [30] Blanz, V., & Vetter, T. (2003). Face recognition based on fitting a 3D morphable model. IEEE Transactions on pattern analysis and machine intelligence, 25(9), 1063-1074.
- [31] Lu, J., Plataniotis, K. N., & Venetsanopoulos, A. N. (2003). Face recognition using LDA-based algorithms. IEEE transactions on neural networks, 14(1), 195-200.
- [32] Ansari, A. N., & Abdel-Mottaleb, M. (2003, July). 3D face modeling using two views and a generic face model with application to 3D face recognition. In Advanced Video and Signal Based Surveillance, 2003. Proceedings. IEEE Conference on (pp. 37-44). IEEE.
- [33] Cook, J., Chandran, V., Sridharan, S., & Fookes, C. (2004, September). Face recognition from 3d data using iterative closest point algorithm and gaussian mixture models. In 3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004. Proceedings. 2nd International Symposium on(pp. 502-509). IEEE.
- [34] Irfanoglu, M. O., Gokberk, B., & Akarun, L. (2004, August). 3D shape-based face recognition using automatically registered facial surfaces. In Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on (Vol. 4, pp. 183-186). IEEE.
- [35] Medioni, G., & Waupotitsch, R. (2003, October). Face recognition and modeling in 3D. In IEEE Intl Workshop on Analysis and Modeling of Faces and Gestures (AMFG 2003)(Vol. 232233).
- [36] Lu, X., Colbry, D., & Jain, A. K. (2004, August). Three-dimensional model based face recognition. In null (pp. 362-366). IEEE.
- [37] Wang, Y., Pan, G., Wu, Z., Han, S. (2004, June). Sphere-spin-image: A viewpoint-invariant surface representation for 3D face recognition. In International Conference on Computational Science (pp. 427-434). Springer, Berlin, Heidelberg.
- [38] Chang, K. I., Bowyer, K. W., & Flynn, P. J. (2005). An evaluation of multimodal 2D+ 3D face biometrics. IEEE transactions on pattern analysis and machine intelligence, 27(4), 619-624.
- [39] Papatheodorou, T., & Rueckert, D. (2004, May). Evaluation of automatic 4D face recognition using surface and texture registration. In Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on(pp. 321-326). IEEE.

- [40] Duy and D.P Helmbold. Boosting methods for regression. Technical report, Department of Computer Science, University of Santa Cruz, 2000