ABSTRACT

Face Recognition is a computer application that is capable of detecting, tracking, identifying or verifying human faces from an image or video captured using a digital camera. Although lot of progress has been made in domain of face detection and recognition for security, identification and attendance purpose, but still there are issues hindering the progress to reach or surpass human level accuracy. These issues are variations in human facial_appearance such as; varying lighting condition, noise in face images, scale, pose etc. This research paper presents a new method using Local Binary Pattern (LBP) algorithm combined with advanced image processing techniques such as Contrast Adjustment, Bilateral Filter, Histogram Equalization and Image Blending to address some of the issues hampering face recognition accuracy so as to improve the LBP codes, thus improve the accuracy of the overall face recognition system.

Our experiment results show that our method is very accurate, reliable and robust for face recognition system that can be practically implemented in real-life environment as an automatic attendance management

system. Face recognition systems have gained immense prominence in recent years, becoming a critical component in various applications such as security, surveillance, authentication, and human-computer interaction. This abstract provides a comprehensive overview of the key advancements in face recognition technology, encompassing both the theoretical underpinnings and practical implementations.

The foundation of face recognition lies in the extraction and analysis of unique facial features, which are then used to identify or verify individuals. Early face recognition systems primarily relied on traditional methods, such as eigenface analysis and template matching. However, recent strides in artificial intelligence, particularly in deep learning, have revolutionized the field.

Deep learning approaches, especially Convolutional Neural Networks (CNNs), have demonstrated remarkable success in extracting intricate facial features and learning robust representations. This abstract delves into the architecture and training processes of these networks, emphasizing their ability to handle variations in pose, lighting conditions, and facial expressions, thus enhancing the accuracy and reliability of face recognition systems.

One significant challenge in face recognition is the need for large annotated datasets for training deep learning models effectively. Transfer learning and data augmentation techniques have emerged as

effective strategies to mitigate this challenge, allowing models to leverage pre-trained knowledge on large datasets and adapt to specific tasks with limited labeled data.

Ethical considerations and concerns related to privacy have become paramount in the deployment of face recognition systems. This abstract explores the ongoing discourse surrounding the responsible use of facial recognition technology, addressing issues such as bias, transparency, and the potential misuse of this technology in surveillance applications. The development of ethical guidelines and regulations to govern the deployment of face recognition systems is crucial to ensuring fair and responsible use.

Beyond traditional 2D face recognition, the advent of 3D face recognition systems has brought about a paradigm shift in the field. This abstract discusses the principles behind 3D face recognition, highlighting its advantages in handling pose variations and providing additional depth information for improved accuracy. The integration of depth-sensing technologies, such as time-of-flight cameras, has paved the way for robust 3D face recognition in various real-world scenarios.

Furthermore, the fusion of multimodal biometrics, combining facial recognition with other biometric modalities such as fingerprints or iris scans, has shown promise in enhancing overall security and reducing vulnerability to spoofing attacks. This abstract explores the synergies between different biometric modalities and their applications in multifactor authentication systems.

In conclusion, this abstract provides a panoramic view of the evolution of face recognition systems, from classical methods to the latest advancements in deep learning, ethical considerations, 3D face recognition, and multimodal biometrics. As face recognition technology continues to mature, its widespread adoption necessitates a balanced approach that considers technological advancements, ethical implications, and regulatory frameworks to ensure its responsible and equitable use in society.

CONTENT

CHAPTER 1

INTRODUCTION

In the vast landscape of technological innovation, facial recognition has emerged as a transformative force with profound implications across diverse sectors, ranging from security and surveillance to consumer electronics and personalized services. The ability of machines to identify and authenticate individuals based on their unique facial features has captivated the imagination of researchers, developers, and society at large. This introduction seeks to unravel the evolution of facial recognition technology, from its nascent stages to the cutting-edge advancements that define its current state.

Historical Perspectives:

The roots of facial recognition can be traced back to the early endeavors in computer vision and pattern recognition. Early systems, rooted in traditional methods, grappled with limited success due to challenges posed by variations in lighting, pose, and facial expressions. As technological capabilities evolved, so did the aspirations of researchers to develop more robust and accurate facial recognition systems.

The Advent of Deep Learning:

A paradigm shift occurred with the rise of deep learning, and particularly Convolutional Neural Networks (CNNs), which revolutionized the field of facial recognition. This section of the introduction delves into the fundamental principles of deep learning, exploring how neural networks, inspired by the human brain, can automatically learn intricate facial features and representations. The shift towards end-to-end learning marked a significant departure from handcrafted feature extraction, paving the way for improved performance and adaptability.

Challenges and Breakthroughs:

Despite the progress brought about by deep learning, challenges persisted. The need for large labeled datasets for training became a bottleneck, leading to the exploration of transfer learning and data augmentation techniques. This section discusses how researchers overcame these challenges, highlighting breakthroughs that contributed to the development of more robust and versatile facial recognition systems.

Ethical Considerations and Privacy Concerns:

As facial recognition technology gained widespread adoption, ethical concerns and privacy issues took center stage. This part of the introduction delves into the ethical implications of facial recognition, addressing issues of bias, transparency, and the potential for misuse in surveillance. The public discourse on balancing innovation with responsible and ethical deployment is crucial for guiding the future trajectory of facial recognition technology.

The Emergence of 3D Face Recognition:

Recognizing the limitations of 2D face recognition, researchers turned to three-dimensional approaches. This section explores the principles behind 3D face recognition, emphasizing how depth information adds a new dimension to the accuracy and reliability of identification. The integration of 3D sensing technologies, such as time-of-flight cameras, has opened up new possibilities for facial recognition in diverse and challenging environments.

Multimodal Biometrics Integration:

Building on the notion of enhancing security, the integration of facial recognition with other biometric modalities, such as fingerprints or iris scans, has gained prominence. This section explores the synergies between different biometric methods and their potential in creating robust multifactor authentication systems, offering heightened security and resilience against spoofing attacks.

In conclusion, this introduction sets the stage for a comprehensive exploration of facial recognition technology. From its historical roots to the latest breakthroughs, ethical considerations, and the integration of 3D and multimodal approaches, facial recognition stands at the intersection of technological prowess and societal impact. As we navigate the intricacies of this rapidly advancing field, it becomes imperative to strike a balance between innovation and responsibility, ensuring that facial recognition technology serves humanity ethically and equitably.

General Overview: - Face Recognition is a computer application that is capable of detecting, tracking, identifying or verifying human faces from an image or video captured using a digital camera.

1.2 Project scope:

Facial recognition to uniquely identify individuals during user onboarding or logins as well as strengthen user authentication activity. Mobile and personal devices also commonly use face analyzer technology for device security.

1.3 Project Features:

- **The main features of the Face recognition system are**

- **Distance between the eyes.**
- **Distance from the forehead to the chin.**
- **Distance between the nose and mouth.**
- **Depth of the eye sockets.**
- **Shape of the cheekbones.**

WORK DONE IN RELATED AREA

Research and development in the field of face recognition have been extensive, covering a broad spectrum of areas such as computer vision, image processing, machine learning, and biometrics. The following highlights key areas of work and advancements in the domain of face recognition:

1. Traditional Methods: Early work in face recognition focused on traditional methods, including eigenface analysis, template matching, and Principal Component Analysis (PCA). These methods laid the groundwork for subsequent developments by providing a foundation for understanding facial feature extraction and representation.
2. Feature Extraction Techniques: The exploration of feature extraction techniques has been a crucial aspect of face recognition research. Local Binary Patterns (LBP), Gabor filters, and Histograms of Oriented Gradients (HOG) are examples of techniques that have been employed to capture distinctive facial features, improving recognition accuracy.
3. Deep Learning Architectures: The advent of deep learning, particularly Convolutional Neural Networks (CNNs), has significantly impacted the field. Deep neural networks can automatically learn hierarchical representations from raw pixel data, allowing for more robust and adaptive face recognition systems. Research has delved into the optimization of CNN architectures, exploring architectures like VGGNet, ResNet, and more recently, transformer-based models.
4. Transfer Learning and Pre-training: Addressing the challenge of limited labeled data, researchers have extensively explored transfer learning and pre-training techniques. Leveraging large datasets and pre-trained models on tasks such as image classification has proven effective in enhancing the performance of face recognition systems, even with smaller datasets.
5. Data Augmentation: Data augmentation has been employed to artificially expand the size of facial recognition datasets, reducing overfitting and improving the generalization capabilities of models. Techniques such as rotation, scaling, and flipping have been applied to augment the training data, enhancing the model's ability to handle variations in pose and lighting conditions.
6. 3D Face Recognition: Traditional 2D face recognition systems face challenges with variations in pose and lighting. To overcome these limitations, researchers have focused on 3D face recognition. Time-of-flight cameras and other depth-sensing technologies have been employed to capture the three-dimensional structure of faces, providing additional information for more accurate identification.
7. Multimodal Biometrics: Combining facial recognition with other biometric modalities, such as fingerprints or iris scans, has been explored to create more robust authentication systems. The fusion of multiple modalities enhances security and reduces the vulnerability of systems to spoofing attacks, offering a more comprehensive approach to biometric identification.
8. Ethical Considerations and Bias Mitigation: As the deployment of face recognition systems became widespread, there has been a growing emphasis on addressing ethical concerns and biases. Researchers have

explored methods to mitigate biases in training data and algorithms, striving for fair and equitable performance across diverse demographic groups.

9. Privacy-Preserving Techniques: Privacy concerns related to facial recognition have led to the development of privacy-preserving techniques. This includes the use of techniques like federated learning, which enables model training across decentralized devices without compromising individual privacy.

10. Real-World Applications: Research efforts have extended beyond theoretical advancements to real-world applications. Face recognition is now integral to various domains, including security and surveillance, mobile devices, access control systems, and human-computer interaction.

In summary, the work in the area of face recognition spans several decades and has seen a progression from traditional methods to the current state-of-the-art deep learning approaches. Ongoing research continues to address challenges, enhance accuracy, and ensure the responsible and ethical deployment of facial recognition technology in diverse applications.

CHAPTER 2

SYSTEM ANALYSIS

2.1 HARDWARE REQUIREMENTS:

- Intel core i5 10 generation is used as a processor because it is fast than other processors an provide reliable and stable and we can run our pc for longtime. By using this processor, we can keep on developing our project without any worries.
- Ram 8GB is used as it will provide fast reading and writing capabilities and will in turn support in processing.
- Web cam

2.2 SOFTWARE REQUIREMENTS:

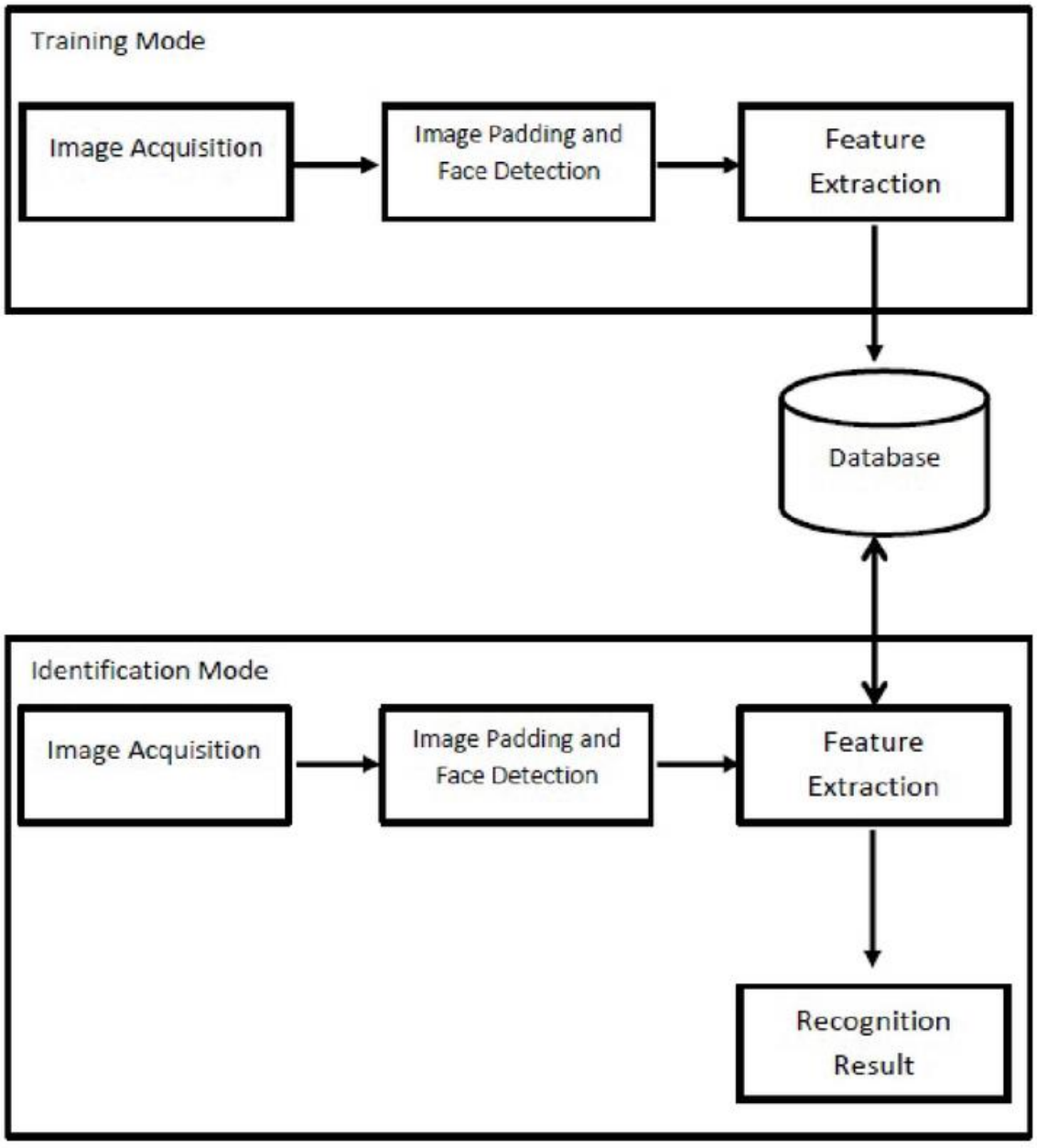
- Operating system- Windows 11 is used as the operating system as it is stable and supports more features and is more user friendly
- Open cv
- Python 3.7 to 3.10
- Face recognition software

CHAPTER 3

SYSTEM DESIGN & SPECIFICATIONS

3.1 HIGH LEVEL DESIGN (HLD)

3.1.1 Flow Graph



CHAPTER 4

CODING

```
from Detector import main_app
from create_classifier import train_classifer
from create_dataset import start_capture
import tkinter as tk
from tkinter import font as tkfont
from tkinter import messagebox,PhotoImage

#from PIL import ImageTk, Image
#from gender_prediction import emotion,ageAndgender
names = set()


class MainUI(tk.Tk):

    def __init__(self, *args, **kwargs):
        tk.Tk.__init__(self, *args, **kwargs)
        global names
        with open("nameslist.txt", "r") as f:
            x = f.read()
            z = x.rstrip().split(" ")
            for i in z:
                names.add(i)
        self.title_font = tkfont.Font(family='Helvetica', size=16, weight="bold")
        self.title("Face Recognizer")
        self.resizable(False, False)
        self.geometry("500x250")
        self.protocol("WM_DELETE_WINDOW", self.on_closing)
        self.active_name = None
        container = tk.Frame(self)
        container.grid(sticky="nsew")
```

```
        container.grid_rowconfigure(0, weight=1)
        container.grid_columnconfigure(0, weight=1)
        self.frames = {}
        for F in (StartPage, PageOne, PageTwo, PageThree, PageFour):
            page_name = F.__name__
            frame = F(parent=container, controller=self)
            self.frames[page_name] = frame
            frame.grid(row=0, column=0, sticky="nsew")
        self.show_frame("StartPage")

    def show_frame(self, page_name):
            frame = self.frames[page_name]
            frame.tkraise()


    def on_closing(self):

        if messagebox.askokcancel("Quit", "Are you sure?"):
            global names
            f =  open("nameslist.txt", "a+")
            for i in names:
                    f.write(i+" ")
            self.destroy()



class StartPage(tk.Frame):


        def __init__(self, parent, controller):
            tk.Frame.__init__(self, parent)
            self.controller = controller
            #load = Image.open("homepagepic.png")
            #load = load.resize((250, 250), Image.ANTIALIAS)
            render = PhotoImage(file='homepagepic.png')
            img = tk.Label(self, image=render)
            img.image = render
            img.grid(row=0, column=1, rowspan=4, sticky="nsew")
```

```
        label = tk.Label(self, text="        Home Page        ", font=self.controller.title_font,fg="#263942")
        label.grid(row=0, sticky="ew")
        button1 = tk.Button(self, text="     Add a User   ", fg="#ffffff", bg="#263942",command=lambda: self.controller.show_frame("PageOne"))
        button2 = tk.Button(self, text="    Check a User   ", fg="#ffffff", bg="#263942",command=lambda: self.controller.show_frame("PageTwo"))
        button3 = tk.Button(self, text="Quit", fg="#263942", bg="#ffffff", command=self.on_closing)
        button1.grid(row=1, column=0, ipady=3, ipadx=7)
        button2.grid(row=2, column=0, ipady=3, ipadx=2)
        button3.grid(row=3, column=0, ipady=3, ipadx=32)


    def on_closing(self):
        if messagebox.askokcancel("Quit", "Are you sure?"):
            global names
            with open("nameslist.txt", "w") as f:
                for i in names:
                    f.write(i + " ")
            self.controller.destroy()



class PageOne(tk.Frame):
    def __init__(self, parent, controller):
        tk.Frame.__init__(self, parent)
        self.controller = controller
        tk.Label(self, text="Enter the name", fg="#263942", font='Helvetica 12 bold').grid(row=0, column=0, pady=10, padx=5)
        self.user_name = tk.Entry(self, borderwidth=3, bg="lightgrey", font='Helvetica 11')
        self.user_name.grid(row=0, column=1, pady=10, padx=10)
        self.buttoncanc = tk.Button(self, text="Cancel", bg="#ffffff", fg="#263942", command=lambda: controller.show_frame("StartPage"))
        self.buttonext = tk.Button(self, text="Next", fg="#ffffff", bg="#263942", command=self.start_training)
        self.buttoncanc.grid(row=1, column=0, pady=10, ipadx=5, ipady=4)
        self.buttonext.grid(row=1, column=1, pady=10, ipadx=5, ipady=4)

    def start_training(self):
```

```
        global names
        if self.user_name.get() == "None":
            messagebox.showerror("Error", "Name cannot be 'None'")
            return
        elif self.user_name.get() in names:
            messagebox.showerror("Error", "User already exists!")
            return
        elif len(self.user_name.get()) == 0:
            messagebox.showerror("Error", "Name cannot be empty!")
            return
        name = self.user_name.get()
        names.add(name)
        self.controller.active_name = name
        self.controller.frames["PageTwo"].refresh_names()
        self.controller.show_frame("PageThree")


class PageTwo(tk.Frame):

    def __init__(self, parent, controller):
        tk.Frame.__init__(self, parent)
        global names
        self.controller = controller
        tk.Label(self, text="Select user", fg="#263942", font='Helvetica 12 bold').grid(row=0, column=0, padx=10, pady=10)
        self.buttoncanc = tk.Button(self, text="Cancel", command=lambda: controller.show_frame("StartPage"), bg="#ffffff", fg="#263942")
        self.menuvar = tk.StringVar(self)
        self.dropdown = tk.OptionMenu(self, self.menuvar, *names)
        self.dropdown.config(bg="lightgrey")
        self.dropdown["menu"].config(bg="lightgrey")
        self.buttonext = tk.Button(self, text="Next", command=self.nextfoo, fg="#ffffff", bg="#263942")
        self.dropdown.grid(row=0, column=1, ipadx=8, padx=10, pady=10)
        self.buttoncanc.grid(row=1, ipadx=5, ipady=4, column=0, pady=10)
        self.buttonext.grid(row=1, ipadx=5, ipady=4, column=1, pady=10)
```

```
    def nextfoo(self):
        if self.menuvar.get() == "None":
            messagebox.showerror("ERROR", "Name cannot be 'None'")
            return
        self.controller.active_name = self.menuvar.get()
        self.controller.show_frame("PageFour")

    def refresh_names(self):
        global names
        self.menuvar.set('')
        self.dropdown['menu'].delete(0, 'end')
        for name in names:
            self.dropdown['menu'].add_command(label=name, command=tk._setit(self.menuvar, name))

class PageThree(tk.Frame):

    def __init__(self, parent, controller):
        tk.Frame.__init__(self, parent)
        self.controller = controller
        self.numimglabel = tk.Label(self, text="Number of images captured = 0", font='Helvetica 12 bold',
fg="#263942")
        self.numimglabel.grid(row=0, column=0, columnspan=2, sticky="ew", pady=10)
        self.capturebutton = tk.Button(self, text="Capture Data Set", fg="#ffffff", bg="#263942",
command=self.capimg)
        self.trainbutton = tk.Button(self, text="Train The Model", fg="#ffffff",
bg="#263942",command=self.trainmodel)
        self.capturebutton.grid(row=1, column=0, ipadx=5, ipady=4, padx=10, pady=20)
        self.trainbutton.grid(row=1, column=1, ipadx=5, ipady=4, padx=10, pady=20)

    def capimg(self):s
        self.numimglabel.config(text=str("Captured Images = 0 "))
        messagebox.showinfo("INSTRUCTIONS", "We will Capture 100 pic of your Face.")
        x = start_capture(self.controller.active_name)
        self.controller.num_of_images = x
        self.numimglabel.config(text=str("Number of images captured = "+str(x)))
```

```
    def trainmodel(self):
        if self.controller.num_of_images < 100:
            messagebox.showerror("ERROR", "No enough Data, Capture at least 100 images!")
            return
        train_classifer(self.controller.active_name)
        messagebox.showinfo("SUCCESS", "The modele has been successfully trained!")
        self.controller.show_frame("PageFour")


class PageFour(tk.Frame):

    def __init__(self, parent, controller):
        tk.Frame.__init__(self, parent)
        self.controller = controller

        label = tk.Label(self, text="Face Recognition", font='Helvetica 16 bold')
        label.grid(row=0,column=0, sticky="ew")
        button1 = tk.Button(self, text="Face Recognition", command=self.openwebcam, fg="#ffffff", bg="#263942")
        #button2 = tk.Button(self, text="Emotion Detection", command=self.emot, fg="#ffffff", bg="#263942")
        #button3 = tk.Button(self, text="Gender and Age Prediction", command=self.gender_age_pred, fg="#ffffff", bg="#263942")
        button4 = tk.Button(self, text="Go to Home Page", command=lambda: self.controller.show_frame("StartPage"), bg="#ffffff", fg="#263942")
        button1.grid(row=1,column=0, sticky="ew", ipadx=5, ipady=4, padx=10, pady=10)
        #button2.grid(row=1,column=1, sticky="ew", ipadx=5, ipady=4, padx=10, pady=10)
        #button3.grid(row=2,column=0, sticky="ew", ipadx=5, ipady=4, padx=10, pady=10)
        button4.grid(row=1,column=1, sticky="ew", ipadx=5, ipady=4, padx=10, pady=10)

    def openwebcam(self):
        main_app(self.controller.active_name)
    #def gender_age_pred(self):
     #  ageAndgender()
    #def emot(self):
     #   emotion()
```

```
app = MainUI()
app.iconphoto(False, tk.PhotoImage(file='icon.ico'))
app.mainloop()
```

create_classifier.py

```
import numpy as np
from PIL import Image
import os, cv2



# Method to train custom classifier to recognize face
def train_classifer(name):
    # Read all the images in custom data-set
    path = os.path.join(os.getcwd()+"/data/"+name+"/")

    faces = []
    ids = []
    labels = []
    pictures = {}


    # Store images in a numpy format and ids of the user on the same index in imageNp and id lists

    for root,dirs,files in os.walk(path):
            pictures = files


    for pic in pictures :

            imgpath = path+pic
```

```
            img = Image.open(imgpath).convert('L')
            imageNp = np.array(img, 'uint8')
            id = int(pic.split(name)[0])
            #names[name].append(id)
            faces.append(imageNp)
            ids.append(id)

    ids = np.array(ids)

    #Train and save classifier
    clf = cv2.face.LBPHFaceRecognizer_create()
    clf.train(faces, ids)
    clf.write("./data/classifiers/"+name+"_classifier.xml"
```

Create_dataset.py

```
import cv2
import os


def start_capture(name):
        path = "./data/" + name
        num_of_images = 0
        detector = cv2.CascadeClassifier("./data/haarcascade_frontalface_default.xml")
        try:
            os.makedirs(path)
        except:
```

```
        print('Directory Already Created')
    vid = cv2.VideoCapture(0)
    while True:

        ret, img = vid.read()
        new_img = None
        grayimg = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        face = detector.detectMultiScale(image=grayimg, scaleFactor=1.1, minNeighbors=5)
        for x, y, w, h in face:
         cv2.rectangle(img, (x, y), (x+w, y+h), (0, 0, 0), 2)

  cv2.putText(img, "Face Detected", (x, y-5), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0, 0, 255))
          cv2.putText(img, str(str(num_of_images)+" images captured"), (x, y+h+20), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0, 0, 255))
           new_img = img[y:y+h, x:x+w]

        cv2.imshow("FaceDetection", img)
        key = cv2.waitKey(1) & 0xFF



try :
           cv2.imwrite(str(path+"/"+str(num_of_images)+name+".jpg"), new_img)
           num_of_images += 1
```

```
        except :


            pass
        if key == ord("q") or key == 27 or num_of_images > 100:
            break
    cv2.destroyAllWindows()
    return num_of_images
```

Detector

```
import cv2
from time import sleep
from PIL import Image


def main_app(name):


    face_cascade = cv2.CascadeClassifier('./data/haarcascade_frontalface_default.xml')
    recognizer = cv2.face.LBPHFaceRecognizer_create()
    recognizer.read(f"./data/classifiers/{name}_classifier.xml")
    cap = cv2.VideoCapture(0)
    pred = 0
    while True:
        ret, frame = cap.read()
```

```
#default_img = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

faces = face_cascade.detectMultiScale(gray,1.3,5)


for (x,y,w,h) in faces:


    roi_gray = gray[y:y+h,x:x+w]


    id,confidence = recognizer.predict(roi_gray)

    confidence = 100 - int(confidence)

    pred = 0

    if confidence > 50:

        #if u want to print confidence level

                #confidence = 100 - int(confidence)

                pred += +1

                text = name.upper()


   font = cv2.FONT_HERSHEY_PLAIN

                frame = cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)

                frame = cv2.putText(frame, text, (x, y-4), font, 1, (0, 255, 0), 1, cv2.LINE_AA)


    else:

                pred += -1
```

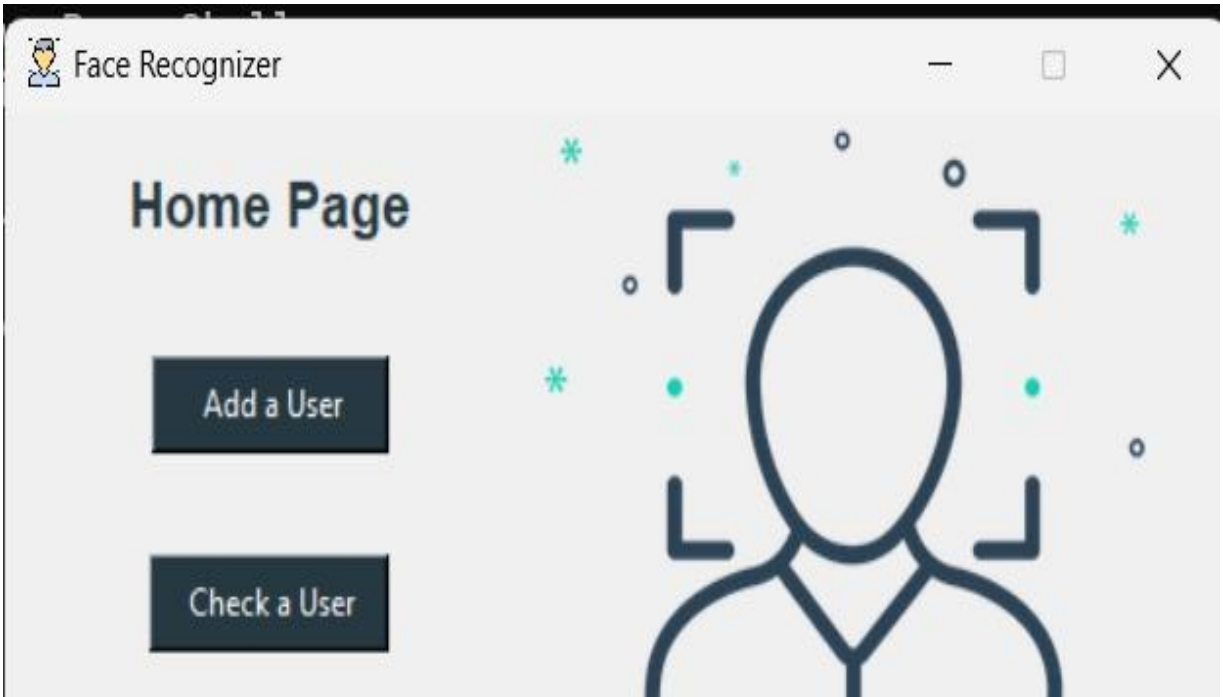
```
            text = "UnknownFace"
            font = cv2.FONT_HERSHEY_PLAIN
            frame = cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 0, 255), 2)
            frame = cv2.putText(frame, text, (x, y-4), font, 1, (0, 0,255), 1, cv2.LINE_AA)

cv2.imshow("image", frame)

if cv2.waitKey(20) & 0xFF == ord('q'):
    print(pred)
    if pred > 0 :
        dim =(124,124)
        img = cv2.imread(f".\\data\\{name}\\{pred}{name}.jpg", cv2.IMREAD_UNCHANGED)
        resized = cv2.resize(img, dim, interpolation = cv2.INTER_AREA)
        cv2.imwrite(f".\\data\\{name}\\50{name}.jpg", resized)
        Image1 = Image.open(f".\\2.png")

        # make a copy the image so that the
        # original image does not get affected

        Image1copy = Image1.copy()
        Image2 = Image.open(f".\\data\\{name}\\50{name}.jpg")
        Image2copy = Image2.copy()
```

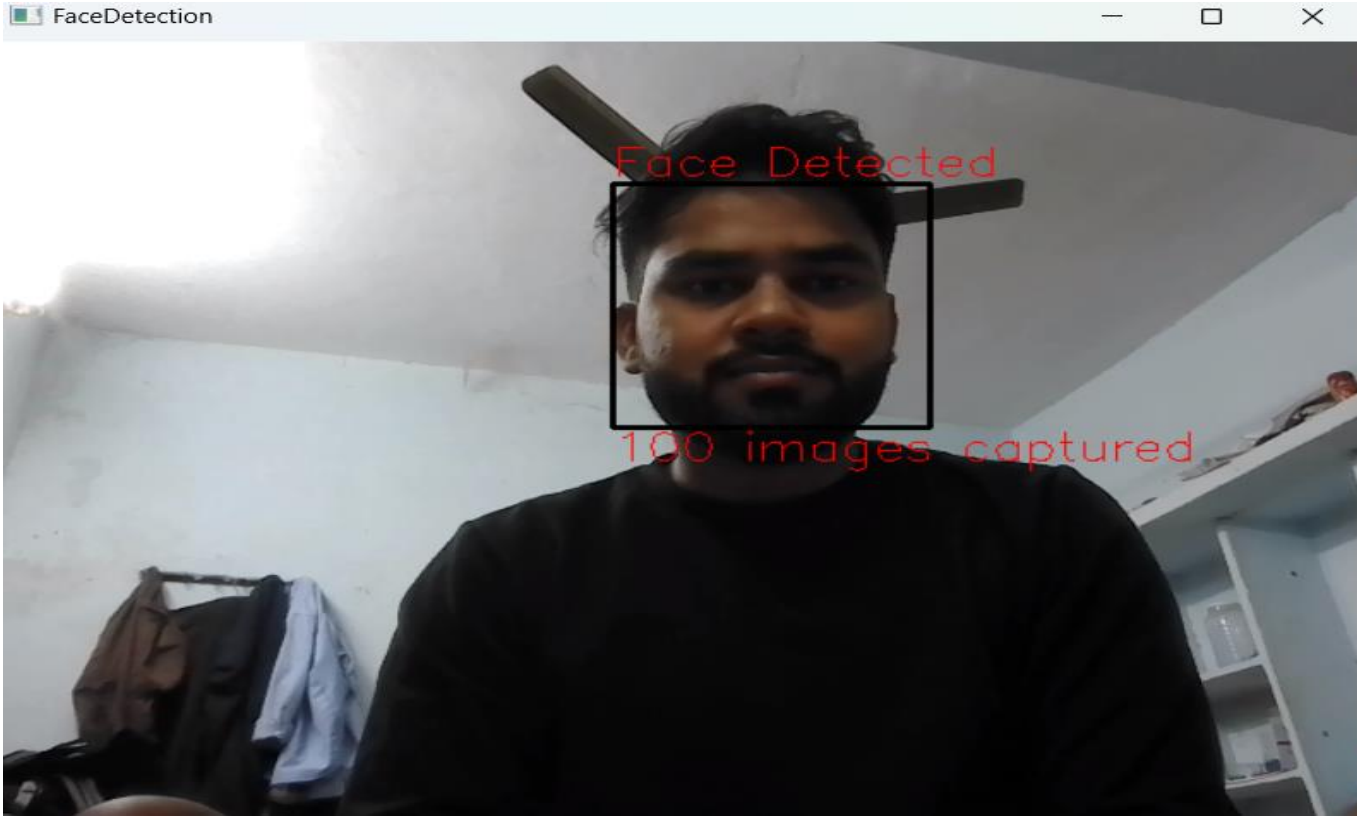
```
            # paste image giving dimensions
            Image1copy.paste(Image2copy, (195, 114))

            # save the image
            Image1copy.save("end.png")
            frame = cv2.imread("end.png", 1)

            cv2.imshow("Result",frame)
            cv2.waitKey(5000)
        break

cap.release()
cv2.destroyAllWindows()
```

CHAPTER 5

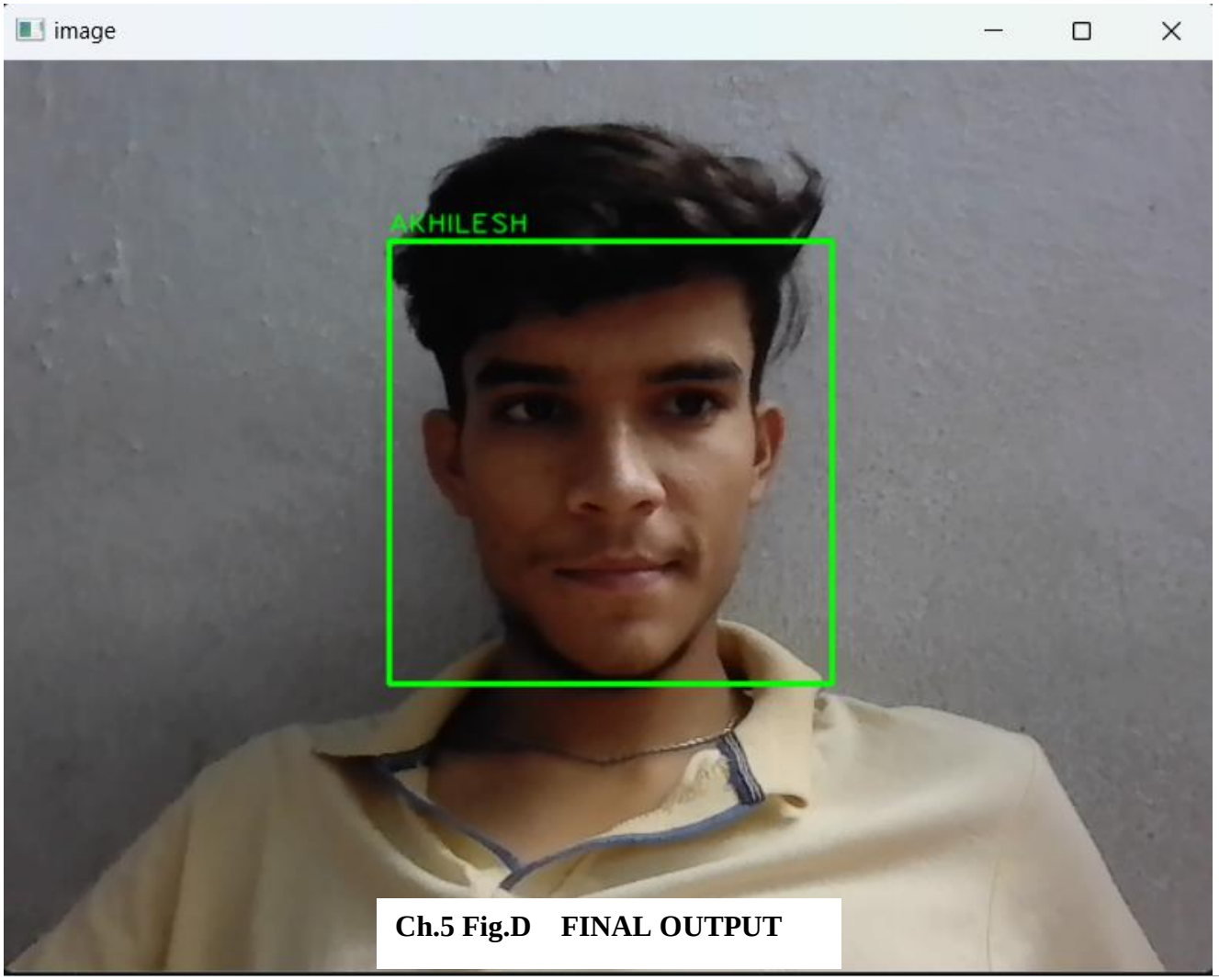
TESTING

Screen Shorts





Ch.5 Fig.C COLLECTING DATASETS FROM WEBCAM



Ch.5 Fig.D FINAL OUTPUT

CHAPTER 6

CONCLUSION & LIMITATIONS

Face recognition technology has come a long way in the last twenty years. Today, machines are able to automatically verify identity information for secure transactions, for surveillance and security tasks, and for access control to buildings etc. These applications usually work in controlled environments and recognition algorithms can take advantage of the environmental constraints to obtain high recognition accuracy. However, next generation face recognition systems are going to have widespread application in smart environments -- where computers and machines are more like helpful assistants.

To achieve this goal computers must be able to reliably identify nearby people in a manner that fits naturally within the pattern of normal human interactions. They must not require special interactions and must conform to human intuitions about when recognition is likely. This implies that future smart environments should use the same modalities as humans, and have approximately the same limitations. These goals now appear in reach -- however, substantial research remains to be done in making person recognition technology work reliably, in widely varying conditions using information from single or multiple modalities.

CHAPTER 7

REFERENCE/BIBLIOGRAPHY

https://pypi.org/project/opencv-python/

https://youtu.be/3EBdT-0gvu8

https://techvidvan.com/tutorials/face-recognition-project-python-opencv/

https://stackoverflow.com/