BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI
(Rajasthan)

A STUDY-ORIENTED PROJECT ON:

Intoxication Detection

Complied by
Garvit Singhal
(2021A7PS2226P)
Sanjeev Mallick
(2021A7PS2217P)

Complied under the guidance of
Prof. Ashutosh Bhatia and Prof. Kamlesh Tiwari
Department of Computer Science & Information Systems, Birla Institute of Technology and Science, Pilani

Acknowledgements

We would like to express my utmost gratitude to **Dr. Ashutosh Bhatia and Dr. Kamlesh Tiwari** for providing me with this opportunity and the requisite knowledge to work on this project and guiding me through every step of it. I have learnt a significant amount of knowledge after working on this project, none of which could have been possible without the constant help and guidance I have received from him.

I would also like to thank **Dr. Navneet Goyal**, Head of Department, Department of Computer Science & Information Systems, for letting me opt for this study-oriented project course as a disciplinary elective under my discipline B.E. Computer Science.

FACE CAPTURE WORKING MODEL

This report describes the working of face capture model designed using python programming language,Graphical user interface using Tkinter and Computer vision.

Task-The main task is to capture faces of an individual using different cameras (Three cameras are used in this model which is used to capture face at different angles).These photos or videos can be captured and are stored in 2 different directories(“photos” and “videos”) which can be used for further use of the data.

Implementation-Below is the implementation using code snippets

```
import os
import cv2
import tkinter as tk
from PIL import Image, ImageTk
import numpy as np
import time
```

These are the necessary import functions necessary for this particular face capture model.

```
class PhotoApp:
    def __init__(self, master):
        self.master = master
        self.master.title("Intoxication Detection Face Capture")

        self.cam_internal = cv2.VideoCapture(0)
        self.cam_external1 = cv2.VideoCapture(2)
        self.cam_external2 = cv2.VideoCapture(4)

        ret_internal, frame_internal = self.cam_internal.read()
        ret_external1, frame_external1 = self.cam_external1.read()
        ret_external2, frame_external2 = self.cam_external2.read()

        if ret_internal:
            self.window_width_internal = frame_internal.shape[1]
            self.window_height_internal = frame_internal.shape[0]
        else:
            self.window_width_internal = 1200
            self.window_height_internal = 800

        if ret_external1:
            self.window_width_external1 = frame_external1.shape[1]
            self.window_height_external1 = frame_external1.shape[0]
        else:
            self.window_width_external1 = 800
            self.window_height_external1 = 600

        if ret_external2:
            self.window_width_external2 = frame_external2.shape[1]
            self.window_height_external2 = frame_external2.shape[0]
        else:
            self.window_width_external2 = 800
            self.window_height_external2 = 600

        self.master.geometry("{}x{}".format(max(self.window_width_internal, self.window_width_external1, self.window_width_external2) * 3,
                                            max(self.window_height_internal, self.window_height_external1, self.window_height_external2)))
```

This part of the code shown in the picture above is responsible for initializing the PhotoApp class, setting up the GUI window, and configuring the video capture from three different cameras (internal, external1, and external2).

```
self.button_frame = tk.Frame(self.master, bg="white")
self.button_frame.pack(side=tk.BOTTOM, fill=tk.X)

self.photo_button_internal = tk.Button(self.button_frame, text="Screenshot (middle)", command=self.take_photo_internal, bg="gray", fg="black")
self.photo_button_internal.pack(side=tk.LEFT, expand=True, fill=tk.X, padx=(10, 5))

self.photo_button_external1 = tk.Button(self.button_frame, text="Screenshot (left)", command=self.take_photo_external1, bg="gray", fg="black")
self.photo_button_external1.pack(side=tk.LEFT, expand=True, fill=tk.X, padx=5)

self.photo_button_external2 = tk.Button(self.button_frame, text="Screenshot (right)", command=self.take_photo_external2, bg="gray", fg="black")
self.photo_button_external2.pack(side=tk.LEFT, expand=True, fill=tk.X, padx=5)

self.video_button_internal = tk.Button(self.button_frame, text="Recording (middle)", command=self.start_recording_internal, bg="gray", fg="black
self.video_button_internal.pack(side=tk.LEFT, expand=True, fill=tk.X, padx=5)

self.video_button_external1 = tk.Button(self.button_frame, text="Recording (left)", command=self.start_recording_external1, bg="gray", fg="black
self.video_button_external1.pack(side=tk.LEFT, expand=True, fill=tk.X, padx=5)

self.video_button_external2 = tk.Button(self.button_frame, text="Recording (right)", command=self.start_recording_external2, bg="gray", fg="blac
self.video_button_external2.pack(side=tk.LEFT, expand=True, fill=tk.X, padx=5)

self.stop_button = tk.Button(self.button_frame, text="Stop Recording", command=self.stop_recording, bg="gray", fg="black")
self.stop_button.pack(side=tk.LEFT, expand=True, fill=tk.X, padx=5)
self.stop_button.config(state=tk.DISABLED)

self.quit_button = tk.Button(self.button_frame, text="Quit", command=self.quit, bg="gray", fg="black")
self.quit_button.pack(side=tk.RIGHT, expand=True, fill=tk.X, padx=(5, 10))

self.timer_label = tk.Label(self.button_frame, text="", bg="white", fg="black")
self.timer_label.pack(side=tk.RIGHT, padx=5)
```

The above code helps to create buttons for various functionalities like taking screenshot,starting recording,stopping recording and quitting the application.

```python
def update_camera(self):
    ret_internal, frame_internal = self.cam_internal.read()
    ret_external1, frame_external1 = self.cam_external1.read()
    ret_external2, frame_external2 = self.cam_external2.read()

    if ret_internal:
        height_internal, width_internal = frame_internal.shape[:2]
        resized_frame_internal = cv2.resize(frame_internal, (int(width_internal * self.zoom_factor), int(height_internal * self.zoom_factor)))
        resized_frame_internal = cv2.cvtColor(resized_frame_internal, cv2.COLOR_BGR2RGB)

    if ret_external1:
        height_external1, width_external1 = frame_external1.shape[:2]
        resized_frame_external1 = cv2.resize(frame_external1, (int(width_external1 * self.zoom_factor), int(height_external1 * self.zoom_factor)))
        resized_frame_external1 = cv2.cvtColor(resized_frame_external1, cv2.COLOR_BGR2RGB)

    if ret_external2:
        height_external2, width_external2 = frame_external2.shape[:2]
        resized_frame_external2 = cv2.resize(frame_external2, (int(width_external2 * self.zoom_factor), int(height_external2 * self.zoom_factor)))
        resized_frame_external2 = cv2.cvtC (variable) ret_external2: bool 2.COLOR_BGR2RGB)

    if ret_internal and ret_external1 and ret_external2:
        combined_frame = np.hstack((resized_frame_external1, resized_frame_internal, resized_frame_external2))
        photo = ImageTk.PhotoImage(image=Image.fromarray(combined_frame))

        self.canvas.delete("all")
        self.canvas.create_image(0, 0, anchor=tk.NW, image=photo)
        self.canvas.image = photo

    self.master.after(10, self.update_camera)
```

This part of the code above defines the update_camera method within the PhotoApp class. It's responsible for continuously updating the displayed frames on the Tkinter canvas with the latest frames captured from the internal and external cameras.

```python
def take_photo_internal(self):
    ret, frame = self.cam_internal.read()
    if ret:
        if not os.path.exists("photos"):
            os.makedirs("photos")
        img_name = "photos/internal_opencv_frame_{}.jpg".format(self.img_counter)
        cv2.imwrite(img_name, cv2.cvtColor(frame, cv2.COLOR_BGR2RGB))
        print("Internal Screenshot taken")
        self.img_counter += 1

def take_photo_external1(self):
    ret, frame = self.cam_external1.read()
    if ret:
        if not os.path.exists("photos"):
            os.makedirs("photos")
        img_name = "photos/external1_opencv_frame_{}.jpg".format(self.img_counter)
        cv2.imwrite(img_name, cv2.cvtColor(frame, cv2.COLOR_BGR2RGB))
        print("External 1 Screenshot taken")
        self.img_counter += 1

def take_photo_external2(self):
    ret, frame = self.cam_external2.read()
    if ret:
        if not os.path.exists("photos"):
            os.makedirs("photos")
        img_name = "photos/external2_opencv_frame_{}.jpg".format(self.img_counter)
        cv2.imwrite(img_name, cv2.cvtColor(frame, cv2.COLOR_BGR2RGB))
        print("External 2 Screenshot taken")
        self.img_counter += 1
```

These three methods (take_photo_internal, take_photo_external1, take_photo_external2) are responsible for capturing screenshots from each of the three cameras: internal, external1, and external2, respectively.

```python
def start_recording_internal(self):
    if not self.is_recording:
        if not os.path.exists("videos"):
            os.makedirs("videos")
        timestamp = time.strftime("%Y%m%d-%H%M%S")
        video_path = os.path.join("videos", f"internal_recorded_video_{timestamp}.avi")
        self.video_writer = cv2.VideoWriter(video_path, cv2.VideoWriter_fourcc(*'XVID'), 20, (self.window_width_internal, self.window_height_interna
        self.is_recording = True
        self.video_button_internal.config(state=tk.DISABLED)
        self.video_button_external1.config(state=tk.DISABLED)
        self.video_button_external2.config(state=tk.DISABLED)
        self.stop_button.config(state=tk.NORMAL)
        self.start_time = time.time()
        self.max_frames = int(5 * 20)
        self.update_timer()
        print("Internal Video recording started")

def start_recording_external1(self):
    if not self.is_recording:
        if not os.path.exists("videos"):
            os.makedirs("videos")
        timestamp = time.strftime("%Y%m%d-%H%M%S")
        video_path = os.path.join("videos", f"external1_recorded_video_{timestamp}.avi")
        self.video_writer = cv2.VideoWriter(video_path, cv2.VideoWriter_fourcc(*'XVID'), 20, (self.window_width_external1, self.window_height_externa
        self.is_recording = True
        self.video_button_internal.config(state=tk.DISABLED)
        self.video_button_external1.config(state=tk.DISABLED)
        self.video_button_external2.config(state=tk.DISABLED)
        self.stop_button.config(state=tk.NORMAL)
        self.start_time = time.time()
        self.max_frames = int(5 * 20)
        self.update_timer()
        print("External 1 Video recording started")
```

These three methods (start_recording_internal, start_recording_external1, start_recording_external2) are responsible for starting the video recording process for each camera: internal, external1, and external2, respectively.)

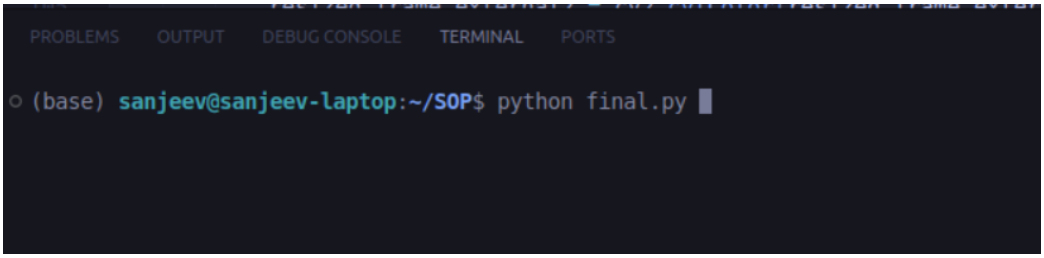
```python
    def stop_recording(self):
        if self.is_recording:
            self.is_recording = False
            self.video_writer.release()
            self.video_button_internal.config(state=tk.NORMAL)
            self.video_button_external1.config(state=tk.NORMAL)
            self.video_button_external2.config(state=tk.NORMAL)
            self.stop_button.config(state=tk.DISABLED)
            print("Video recording stopped")

    def update_timer(self):
        if self.is_recording:
            elapsed_time = int(time.time() - self.start_time)
            minutes = elapsed_time // 60
            seconds = elapsed_time % 60
            self.timer_label.config(text="Recording Time: {:02d}:{:02d}".format(minutes, seconds))
            self.master.after(1000, self.update_timer)

    def quit(self):
        self.cam_internal.release()
        self.cam_external1.release()
        self.cam_external2.release()
        self.master.destroy()

def main():
    root = tk.Tk()
    app = PhotoApp(root)
    root.mainloop()

if __name__ == "__main__":
    main()
```

These methods as shown in the figure above handle stopping video recording, updating the recording timer, quitting the application, and the main execution flow of

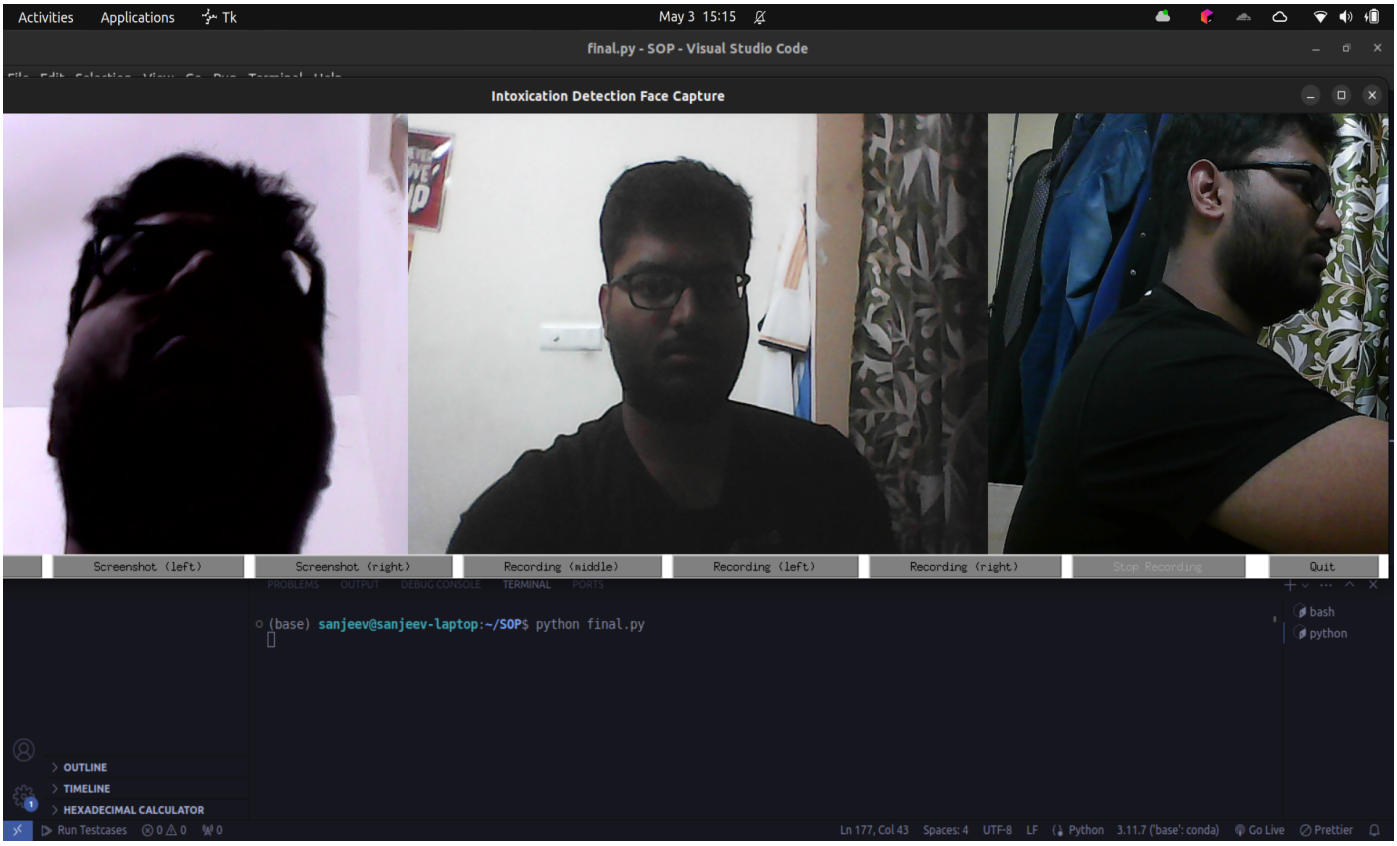
the program.

Demonstration-To run the application type the following in the terminal as follows

python filename.py



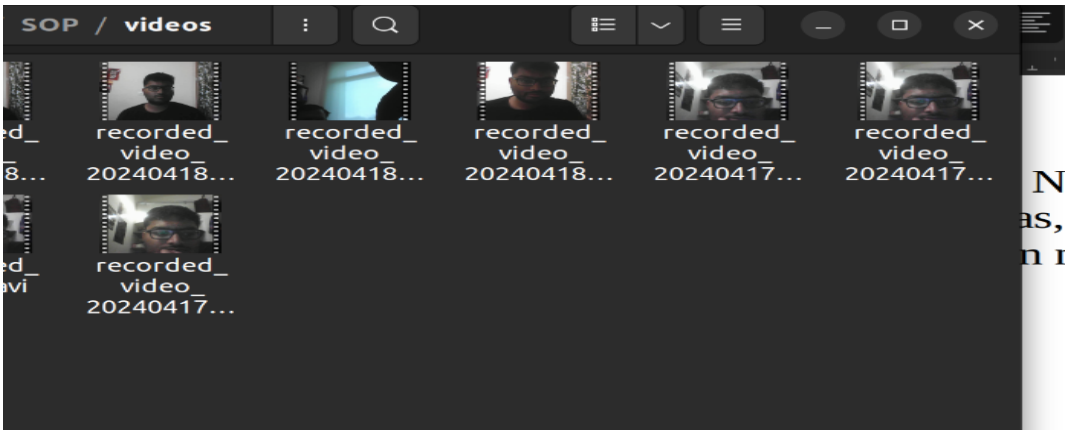
After running the following,we will get a screen as shown below →



Here we can see that we can capture the faces using three cameras. Now we can apply various functions like taking screenshots of any of the three cameras,capturing video from any of the three cameras and also stopping the recording when needed. Finally when we are done with our work we can quit.

Below are some samples →

We can retrieve videos from videos directory as show below



Similarly for photos,we can retrieve from photos directory as shown below