GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Sketch and Signature using Object Tracking

A Project Report of Capstone Project – 2

Submitted by

ASHISH KUMAR SINGH

(1613101196 / 16SCSE101560)

in partial fulfillment for the award of the degree

of

Bachelor of Technology

IN

Computer Science and Engineering

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

Under the Supervision of

Mr. Praveen Dominic

Assistant Professor

APRIL/MAY-2020

GALGOTIAS UNIVERSITY

SCHOOL OF COMPUTING AND SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

Certified that this project report **"Sketch and Signature using Object Tracking"** is the bonafide work of **"ASHISH KUMAR SINGH (1613101195)"** who carried out the project work under my supervision.

SIGNATURE OF HEAD

Dr. MUNISH SHABARWAL, PhD (Management), PhD (CS)
Professor & Dean,
School of Computing Science & Engineering

SIGNATURE OF SUPERVISOR

Mr. PRAVEEN DOMINIC,
Assistant Professor
School of Computing Science & Engineering

ABSTRACT

Being able to keep track of daily schedule and notes of information much quickly and easily could be useful. I propose an application “Text Corner” that uses computer webcams to detect and track a pen by object tracking using Contour-based tracking. And movement of the pen is used to understand the user’s writings. By using this application, user should be able to take short notes, write short messages, draw simple sketches with their free hand. The pen’s path is traced in the video frame and this path is used to draw simple sketches, write short messages and signatures. The Image file then can either be saved in drive or sent on mail. This application is developed in Python using OpenCV. Future work on this project is focused on adding new features to this application like adding an AI system that can help dumb people, these user just needed to swing their pen in the air and the AI system will recognize what the user want him to do and perform the desired task. This AI system would respond on the text that user is trying to write in air, rather on speaking the commands or task.

LIST OF TABLES

LIST OF FIGURES

TABLE OF CONTENTS

1.INTRODUCTION

Nowadays the world is moving towards automation and smart world, so an automated and intelligent system that can be used to write text, simple sketching and short messages and send the same on an e-mail or save the same in format either in text or jpg (for saving as image) is becoming a necessity. Think of scenario where, Ram is in office and he wants to take note that as soon as he reaches home in the evening, he has to pay the maintenance bill and have to make a call to one friend of his named “Rahul”. He just needed to start the application and swing the pen in the air and write the notes like “pay the bill” and “call Rahul”. The application will trace the movement of pen and write the same on the screen and send that notes on your mail id. And once he reaches home, he checks his mail in inbox saying about payment and calling and pays the maintenance bill and calls “Rahul” and talk to him. This application can be used for simple sketching and in the situation when the user needed to upload his/her signature in the JPEG format or as Image. This application provides a facility to save the signature just by swinging their signature in the air. This application can be very useful in the community of people who cannot speak. Like in “Google Assistant” or any other “AI Assistant” the user needed to speaks of the thing to be searched in browser. But in this you don’t need to speak of the task. You just need to write in the air of the task or thing you needed to be searched over the internet. The images shown below are some outputs. Figure 1. Shows the written message for payment. Fig. 2 for calling Rahul.

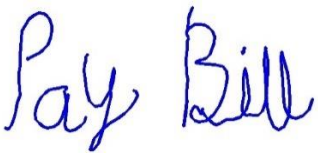

Figure 1: Payment Text Message

Figure 2: Message for call

In today's world major portion of the society do not like to type messages and there are many reasons for this. Some of the reasons are keys of small sizes, little space between keys and bad interface. And results in bad user experience, so Text Corner can be used as an alternate.

Often when we are running late, we take short notes on a piece of paper, sticky notes etc. depending on whatever is available easily and the problem arises when we lose the paper or sticky notes and then we also lose the information. The information can sometime be very critical and important. All this would not have happened if there was facility of internet synchronization. The user can take notes just by swinging pen and the same to be send to his/her e-mail. Think of situation where Ram is on call and want to note the phone number, Ram just needed to write the number in air in front of Text Corner and the phone number can be send on mail as image or text file. Ram can see the number on his mail and when done with it can delete it. Text Corner keeps the backup of information till the user deletes this manually.

As the application works by tracking object by color. Efficiency of Text Corner depends on two factors Background of Pen and intensity of the room. In this application we are tracking pen of "BLUE" color and talking of background of pen if the background is white or any color other than blue it would be easy to track the pen and the efficiency will be high but if the background contains the shades of blue then it will be hard to track the pen and efficiency will be low. And second is the light intensity of the room as object color shades changes in dim or bright light, so to overcome this we are using range of values for identifying blue color. Lower range of HSV color used to detect blue is [35, 174, 179] and Upper range of HSV color used is [140, 255, 255]. And then we track object of blue color which falls between lower and upper range of HSV color.

Contours of the object are identified and then the center points of these contours are computed and stored in a list. A line is drawn between these center points and the result obtained is the text or sketch user wanted. Figure below shows a simple text, that can be written. Some basic sample words output of this application is shown below.

Figure 3: Some basic sample words

2.LITERATURE REVIEW

In [1] Chirag I. Patel and Ripal Patel proposed contour-based object tracking. This involves tracking of boundary contour of a moving object from frame to frame in video. This uses color histogram of object for fast tracking of objects because this is computationally inexpensive. This method is capable of tracking object even if some frames are missed. This method works fine if the object color is little bit of different from the background.

In [2] A Contour-Based Moving Object Detection and Tracking by Masayuki Yokoyama and Tomaso Poggio proposed a fast and robust approach for identification and tracking of object. Their method is based on using lines computed by a gradient-based optical flow and edge detector. Extracted edges by using optical flow and the edge detector are restored as lines, and background lines of the previous frame are subtracted. Contours of objects are obtained by using snakes to clustered lines. Detected objects are tracked.

In [3] ACM Based ROI Extraction for Pedestrian Detection with Partial Occlusion Handling. In this the method uses ACM (Active Contour Model) for locating pedestrian position from frame to frame in video. This uses HOG (Histogram of Oriented Gradients) and LBP (Local Binary Patterns) as features to train a linear SVM. The proposed method uses two level SVM classifier.

In [4] Contour Based Object Tracking. This

proposes tracking of object in images and videos using contours. Using contours centroid of the objects are identified, these center points represent the object. And the path of the object can be traced easily. This uses lower range values for blue as [110,50,150] and upper range value as [130,260,255].

In [5] Using Mobile Phones to Write in Air, this paper proposes the system that uses built-in accelerometer of mobile phones. User had to hold the phone like a pen and try to write in air. The phone identifies the hand gesture as strokes sequence and then checked against a grammar and recognizes the alphabet written in the air.

In [6] Moving Object Tracking Using Object Segmentation. This paper proposes an approach for tracking moving objects in sequence of images or each frame of videos by using object segmentation framework and feature matching functionality.

In [7] for tracking the motion of moving objects. This paper made use of the neural network to track the motion of moving objects recorded in a sequence of video images. Given the motion vector of an arbitrary pixel and color, the neural network determines if the pixel belong to the moving object or not. This method attains high accuracy for object identification.

In [8] for object detection using background subtraction. First, extract the moving objects from the image sequence using background subtraction. Second, select part of the objects for further detected. Third, extract several significant eigen values from the rest objects, which can indicate

the differences of contour between pedestrian and vehicle. Finally, eigen vector is formed and used as the input of the back propagation neural network, the output of which is the detecting result.

In [9] makes use of Kalman filter. Kalman filter model is used to tracking and predicting the trace of an object. Image enhancement is the process of adjusting tracked frame images so that the results are more suitable for display.

3.PROPOSED SYSTEM

Aim of this project is to use object tracking approach for writing, sketching, signature, short messages etc. This provides user the facility to write in air and the same can be send on mail or translated in text and saved as text files. This is helpful for audience who are not interested in typing. This provides user with the color selection menu. This application uses contour-based object tracking. Each frame of video is the input to this application. And some image processing needed to be done on each frame like:

1)flip the frame:

```
frame = cv2.flip(frame, 1)
```

2)Apply Gaussian Blur:

```
blur_frame=cv2.GaussianBlur(frame, (7, 7), 0)
```

3)Convert BGR to HSV format:

```
hsv=cv2.cvtColor(blur_frame, cv2.COLOR_BGR2HSV)
```

Each colored object can be identified in background using thresholding functions. Each color has its own range of lower and upper range of intensity values. And the pixels that have their intensity value in the range of upper and lower range of any color, the pixel will be considered of that color. The approximate lower and upper range of intensity value of some basic color are:

Color	Lower Range	Upper Range
Blue	[35, 174, 179]	[140, 255, 255]
Red	[160, 170, 50]	[179,250,220]
Green	[53,74,160]	[90,147,255]
Yellow	[20,100,100]	[30,255,255]

Table 1: Approximate intensity value of some basic colors.

The object is identified by its color. When the object is detected in the frame and a contour is drawn for that object in this case a blue pen. And that contour is going to represent the edges of object. Once the contour is known, the next thing is to compute the center point for that object i.e., centroid of the object. In order to find center point we have to calculate the first order central moments of the identified object. The formula used for calculation of centroid is:

Centroid (X ,Y)= ?

X = int(M[’m10’]/M[’m00’])

Y = int(M[’m01’]/M[’m00’])

After the centroid co-ordinates are identified, these center points are saved in a list for future work. The center points obtained from each frame is used to draw filled line between them and thus giving the desired output either the written text, short messages or sketch.

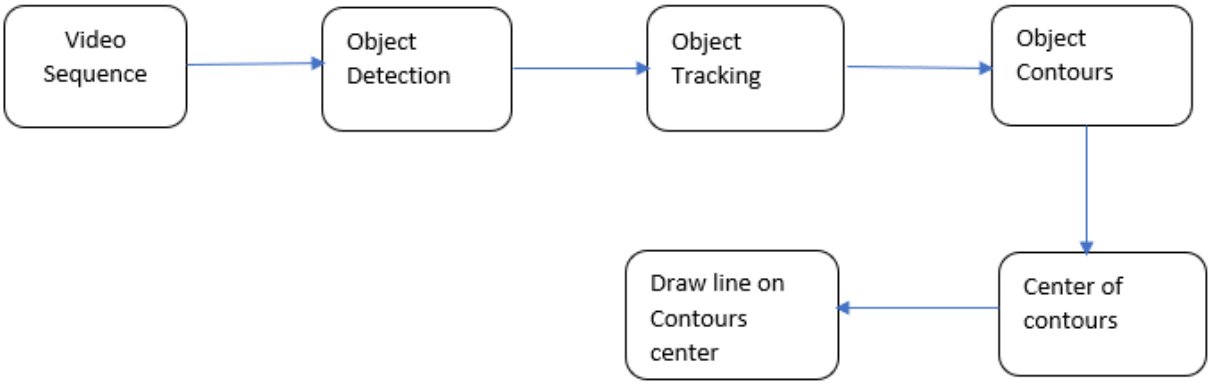

Figure 4: Flow Chart of Proposed Model.

4.IMPLEMENTATION

Text_Corner.py

```
import tkinter
import cv2
import PIL.Image, PIL.ImageTk
from PIL import Image,ImageTk
import time
import math
import numpy as np
import random
from collections import deque
from tkinter import *
from tkinter.ttk import *
```

```
from time import strftime
class App:


    def choosenBlue(self):
        selec=(255,0,0)
        self.pointColor.append(selec)
        print("blue selected")
        return("blue")
    def choosenRed(self):
        selec=(0,0,255)
        self.pointColor.append(selec)
        print("Red selected")
        return("red")
    def choosenGreen(self):
        selec=(0,255,0)
        self.pointColor.append(selec)
        print("Green selected")
        return("gren")
    def choosenYellow(self):
        selec=(0,255,255)
        self.pointColor.append(selec)
        print("Yellow selected")
        return("yellow")
```

```
def choosenBrown(self):
    selec=(0,76,153)
    self.pointColor.append(selec)
    print("Brown selected")
    return("brown")

def __init__(self, window, window_title, video_source=0):
    self.window = window
    self.window.title(window_title)
    self.video_source = video_source
    menubar = Menu(self.window)

    self.pointColor=[(255,0,0),]

    # Adding File Menu and commands
    color1 = Menu(menubar, tearoff = 0)
    menubar.add_cascade(label ='Select Color', menu =color1 )
    color1.add_command(label ='Blue', command =self.choosenBlue)
    color1.add_command(label ='Red', command = self.choosenRed)
    color1.add_command(label ='Green', command =self.choosenGreen)
    color1.add_command(label ='Yellow', command = self.choosenYellow)
    color1.add_command(label ='Brown', command = self.choosenBrown)
```

```
        color1.add_separator()
        color1.add_command(label ='Exit', command = self.window.destroy)
        self.window.config(menu = menubar)
        # open video source (by default this will try to open the computer webcam)
        self.vid = MyVideoCapture(self.video_source)
        # Create a canvas that can fit the above video source size
        self.canvas = tkinter.Canvas(window, width = self.vid.width*2, height = self.vid.height)
        self.canvas.pack()
        self.center_points = deque()
        self.index_i=0
        self.dictPoint={0:(255,0,0),}
        self.img=cv2.imread("art.JPG")
        self.img = cv2.resize(self.img,(640,480))
        # Button that lets the user take a snapshot
        self.btn_snapshot=tkinter.Button(window, text="Snapshot", width=50,
command=self.snapshot)
        self.btn_snapshot.pack(anchor=tkinter.CENTER, expand=True)

        # After it is called once, the update method will be automatically called every delay
milliseconds
        self.delay = 15
        self.update()
```

```
        self.window.mainloop()

    def snapshot(self):
        # Get a the image saved

        cv2.imwrite("Snapshot-" + time.strftime("%d-%m-%Y-%H-%M-%S") + ".jpg",
cv2.cvtColor(self.img, cv2.COLOR_RGB2BGR))



    def update(self):
        # Get a frame from the video source
        ret, frame = self.vid.get_frame()
        frame=cv2.flip(frame,1)

        if ret:
            #photo1 = PIL.ImageTk.PhotoImage(image = PIL.Image.fromarray(frame))
            frame1=cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
            self.photo = PIL.ImageTk.PhotoImage(image = PIL.Image.fromarray(frame1))
            #load = Image.open("art.jpg")
            self.selectColor=(255,255,255)

            self.render = PIL.ImageTk.PhotoImage(image = PIL.Image.fromarray(self.img))
```

```
            self.canvas.create_image(640, 0, image = self.photo, anchor = tkinter.NW)
            self.canvas.create_image(0, 0, image = self.render, anchor = tkinter.NW)
            # Blur the frame a little
            blur_frame = cv2.GaussianBlur(frame, (7, 7), 0)

            # Convert from BGR to HSV color format
            hsv = cv2.cvtColor(blur_frame, cv2.COLOR_BGR2HSV)

            # Define lower and upper range of hsv color to detect color,
            # Blue color in this case
            lower_blue = np.array([35, 174, 179])
            upper_blue = np.array([140, 255, 255])
            mask = cv2.inRange(hsv, lower_blue, upper_blue)
            # Make elliptical kernel
            kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (15, 15))

            # Opening morph(erosion followed by dilation)
            mask = cv2.morphologyEx(mask, cv2.MORPH_OPEN, kernel)

            # Find all contours
            contours, hierarchy = cv2.findContours(mask.copy(), cv2.RETR_LIST,
cv2.CHAIN_APPROX_SIMPLE)[-2:]
            if len(contours) > 0:
```

```
            # Find the biggest contour
            biggest_contour = max(contours, key=cv2.contourArea)

            # Find center of contour and draw filled circle
            moments = cv2.moments(biggest_contour)
            centre_of_contour = (int(moments['m10'] / moments['m00']), int(moments['m01'] /
moments['m00']))
            cv2.circle(frame, centre_of_contour, 5, (0, 0, 255), -1)

            # Bound the contour with circle
            ellipse = cv2.fitEllipse(biggest_contour)
            cv2.ellipse(frame, ellipse, (0, 255, 255), 2)

            # Save the center of contour so we draw line tracking it
            self.center_points.appendleft(centre_of_contour)

        for i in range(1, len(self.center_points)):
            b = random.randint(230, 255)
            g = random.randint(100, 255)
            r = random.randint(100, 255)
            if math.sqrt(((self.center_points[i - 1][0] - self.center_points[i][0]) ** 2) + (
                (self.center_points[i - 1][1] - self.center_points[i][1]) ** 2)) <= 50:
                cv2.line(frame,self.center_points[i - 1], self.center_points[i], (b, g, r), 4)
```

```
                cv2.line(frame1,self.center_points[i - 1], self.center_points[i], (b, g, r), 4)
                cv2.line(self.img, self.center_points[i - 1], self.center_points[i], self.pointColor[-
1], 4)
            #cv2.imshow('original', frame)
            #cv2.imshow('mask', mask)
            self.img=cv2.cvtColor(self.img,cv2.COLOR_RGB2BGR)
            #cv2.imwrite("newart.JPG",img)
            self.photo = PIL.ImageTk.PhotoImage(image = PIL.Image.fromarray(frame1))
            self.render = ImageTk.PhotoImage(image = PIL.Image.fromarray(self.img))
            self.canvas.create_image(640, 0, image =self.photo, anchor = tkinter.NW)
            self.canvas.create_image(0, 0, image = self.render, anchor = tkinter.NW)
            #cv2.imshow('Art',img)
        self.window.after(self.delay, self.update)


class MyVideoCapture:
    def __init__(self, video_source=0):
        # Open the video source
        self.vid = cv2.VideoCapture(video_source)
        if not self.vid.isOpened():
            raise ValueError("Unable to open video source", video_source)

        # Get video source width and height
```

```
            self.width = self.vid.get(cv2.CAP_PROP_FRAME_WIDTH)
            self.height = self.vid.get(cv2.CAP_PROP_FRAME_HEIGHT)

        def get_frame(self):
            if self.vid.isOpened():
                ret, frame = self.vid.read()
                if ret:
                    # Return a boolean success flag and the current frame converted to BGR
                    #return (ret, cv2.cvtColor(frame, cv2.COLOR_BGR2RGB))
                    return (ret,frame)

                else:
                    return (ret, None)
            else:
                return (ret, None)

        # Release the video source when the object is destroyed
        def __del__(self):
            if self.vid.isOpened():
                self.vid.release()

    # Create a window and pass it to the Application object
App(tkinter.Tk(), "Text Corner")
```

5.OUTPUT

The Object Tracking using contour involves several steps and the output from one step is input to another step. The contour identification and path tracking were excellent while testing. And the color selection is easy and free hand drawing response is very good. The image files that meant to be sent on mail was sent successfully in all the attempts. And all the snapshots were saved perfectly.

[1] Start Screen

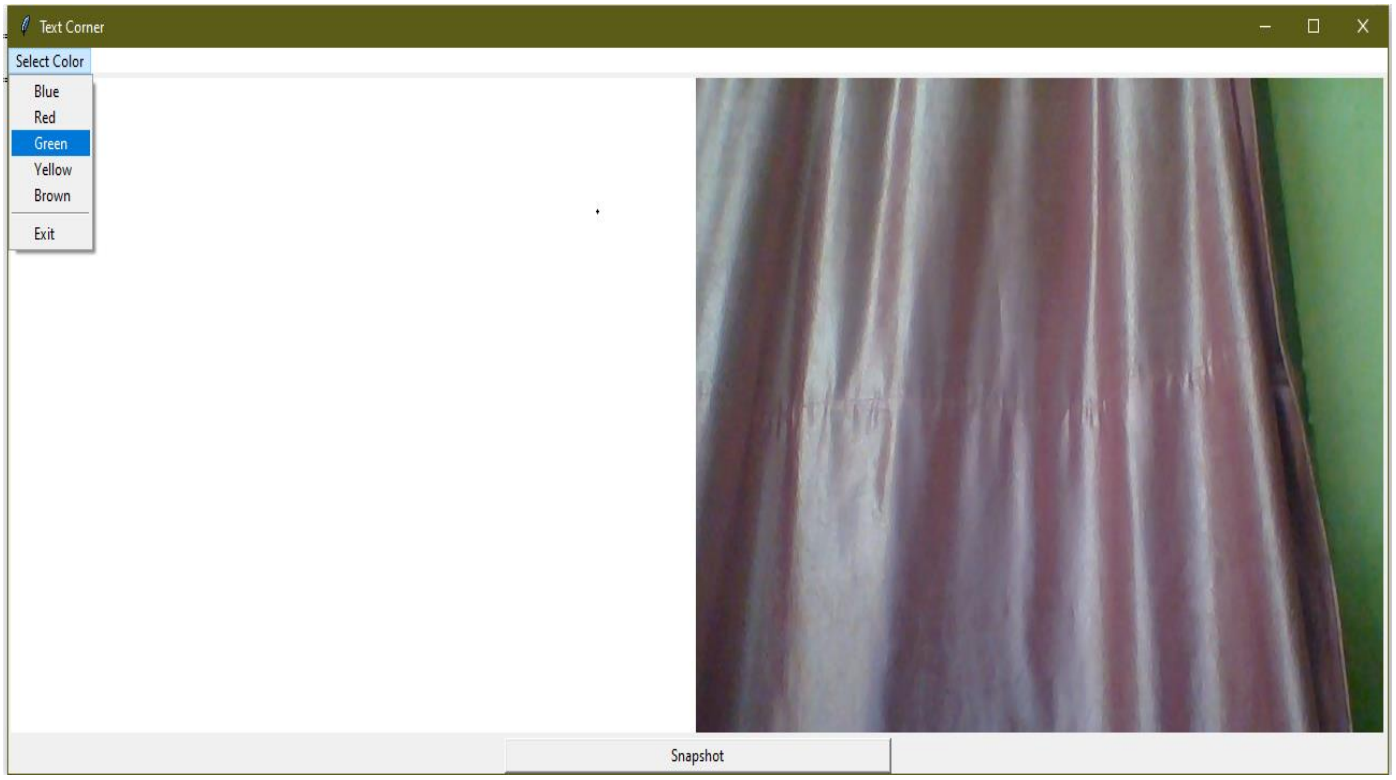

Figure 5: Start Screen of Text Corner Application

[2] Writing word “Ball” on application

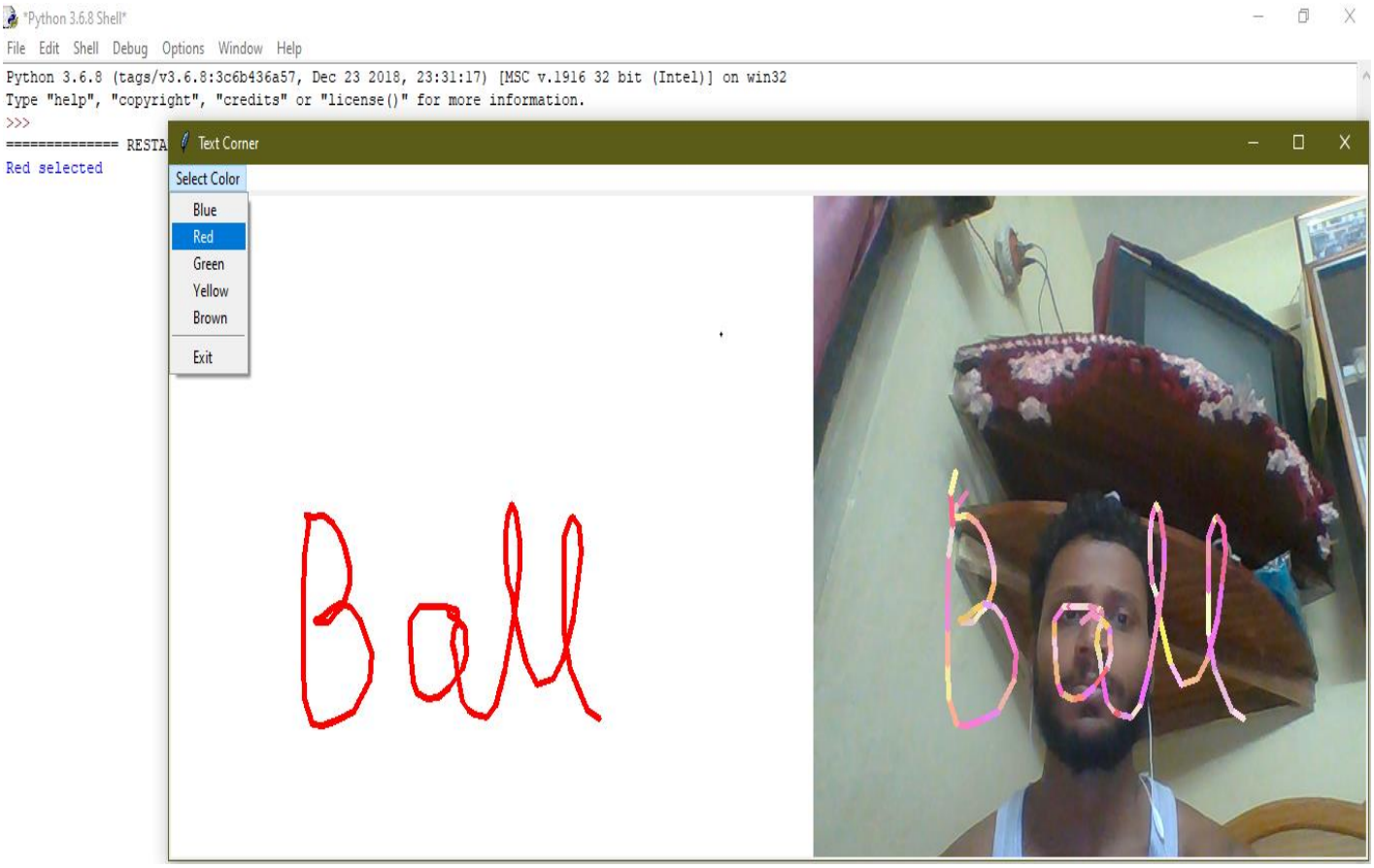

Figure 6: Simple word “Ball” written using application.

[3] Image file saved via “snapshot” button.

Figure 7: Image saved as “Snapshot-02-05-2020-18-17-34.jpg”

6.FUTURE WORK

This application works fine with writing text, sketching, and short messages. But what if we integrate this application with AI assistant. This would be a next level application where user do not need to say anything, they just need to type things like “Hey there, switch on fan” and this too in air by swinging pen like a magical wand and the task would be completed. This would be of much help for those people who are unable to speak, they can simply write out their thoughts in the air or can even interact with an AI without speaking. One more feature that can be added in this is to understand and respond on sign language and write down to text.

7.CONCLUSION

Object Detection and Tracking is an important field in computer vision. The contour-based tracking is very helpful and easy to implement. This application was developed by keeping in mind the comfort of user. The application is programmed in python, using OpenCV library. Python and the libraries included in this are open source and thus cost effective. And this do not have high processor requirements. In testing phase the application was tested positive, as expected.

8.REFERENCES

[1] Chirag I. Patel and Ripal Patel,” Contour Based Object Tracking,” International Journal of Computer and Electrical Engineering, Vol. 4, No. 4, August 2012.

[2] Masayuki Yokoyama and Tomaso Poggio,” A Contour-Based Moving Object Detection and Tracking” published in 2005

[3] Author links open overlay panelViswajith P.Viswanath[a] N. K.Ragesh[b] Madhu S.Nair[a], ”ACM Based ROI Extraction for Pedestrian Detection with Partial Occlusion Handling,” Procedia Computer Science Volume 46, 2015, Pages 45-52.

[4] Jaya P and Geethu Balakrishnan," Contour Based Object Tracking," Jaya P et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (3) , 2014, 4128-4130.

[5] Sandip Agrawal, Ionut Constandache, Shravan Gaonkar, Romit Roy Choudhury, Kevin Cave and Frank DeRuyter," Using Mobile Phones to Write in Air," published in 2011.

[6] Singh S., Dunga S.M., Mandal A.S., Shekhar C., Vohra A. (2010) Moving Object Tracking Using Object Segmen- tation. In: Das V.V., Vijaykumar R. (eds) Information and Communication Technologies. ICT 2010. Communications in Computer and Information Science, vol 101. Springer, Berlin, Heidelberg

[7] Takaya et al," neural network to track the motion of moving objects," Given the motion vector of an arbitrary pixel and colour, the neural network determines if the pixel belong to the moving object or not. published in 2005.

[8] Deng et al.," Back propagation (BP) neural network", published in 2010.

[9] Meenatchi et al.,"Kalman filter" Kalman filter model is used to tracking and predicting the trace of an object. Published in 2014.

[10] A.Yilmaz, O. Javed, and M.Shah, "Object Tracking: A Survey,"ACM Comput. Surv,vol. 38,no. 4, pp.13,2006.

[11] Maria Isabel Ribeiro "Introduction to Kalman Filtering", Institue of Super Tehnico, June 2000.

[12] Hitesh A Patel, Darshak G Thakore,"Moving Object Tracking Using Kalman Filter", International Journal of Computer Science and Mobile Computing,April 2013, pg.326 - 332.

[13] P Dokladal, R Enficiaud, E. Dejnozkova “Contour-based object tracking with gradient-based contour attraction field” EEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP ’04) ,pp. 17-20, 2004