

A
Minor Project Report on
“DRAW IN AIR”

In partial fulfillment of requirements for the degree of
Bachelor of Technology (B. Tech.)
in
Computer Science and Engineering



Submitted by
Ms. Kishita Sureka (170306)

Under the Guidance of
Dr. Sunil Kumar Jangir

SCHOOL OF ENGINEERING AND TECHNOLOGY
Mody University and Science and Technology
Lakshmangarh, Distt. Sikar-332311

December 2020

A C K N O W L E D G E M E N T

In performing Minor Project, I had to take help and guidance of some respected people, who deserves my greatest gratitude. The completion of this project gives me much pleasure. I would like to thank **Dr. A. Sentil**, Dean CET for giving me an opportunity to work on this project. Also, we would like to thank Dr. Sunil Kumar Jangir whose continued support, guidance and vision have helped us in this project.

Kishita Sureka

CERTIFICATE

This is to certify that the minor project report entitled “Draw in Air ”submitted by Ms. Kishita Sureka a partial fulfillment for the requirement of B. Tech. VII Semester examination of the School of Engineering and Technology, Mody University of Science and Technology, Lakshmangarh for the academic session 2019-2020 is an original project work carried out under the supervision and guidance of Dr. Sunil Kumar Jangir has undergone the requisite duration as prescribed by the institution for the project work.

PROJECT GUIDE:

Approval Code: AUT_20_CSE_F12_04

Name: Dr. Sunil Kumar Jangir

Date: 27/12/2020

HEAD OF DEPARTMENT

Signature: Dr. A. Sentil

Name: Dr. A. Sentil

Date: 27/12/2020

EXAMINER-I:

Name: Dr. Praneet Saurabh

Dept: CET

EXAMINER-II

Name: Dr. Ajay Kumar

Dept: CET

ABSTRACT

Now a days a pen tab is the most important requirement and this will help to provide a digital board that can operate using figure. A digital board will also be helpful in better understanding of students. We can draw in air using a simple Data Structure(Queue) for collect pixel and OpenCV for detect object. First we will read video and detect object then pixels on which object is moving collect in queue.

Through this we can provide many features in live camera like colour of different pen, eraser. Different features will also helpful for clear understanding of students and will make teaching easy for teachers. This will do a work similar to a pen tab or we can say that this will be a cheap version of pen tab. Pen tab is not effort able for every teacher so, will be a cheap version of that.

Table of Contents

Sr.no.	Topics	Page no.
1.	Introduction	1
1.1	<i>-Present System</i>	1
1.2	<i>-Proposed System</i>	3
2.	System Design	
2.1	<i>-System flowchart *</i>	5
3.	Hardware and Software detail	
3.1	<i>-Hardware Requirement</i>	8
3.2	<i>-Software Requirement</i>	8
4.	Implementation Work Details	
4.1	<i>-Real life applications</i>	10
4.2	<i>-Data implementation and program execution</i>	14
5.	Source Code/Simulation Code/Commands etc.	19
6.	Input/output Screens/ Model's Photograph	27
7.	Conclusion	
7.1	<i>-Limitations</i>	29
7.2	<i>-Scope for future work</i>	29
8.	Bibliography <i>(Strictly in IEEE style only)</i>	31
9.	Annexures	32
	<i>- Plagiarism Report (Maximum 10-15% excluding code)</i>	

List of Figures

4.1 PEN TAB COPY	10
4.2 MAPPING 3D DRAWING IN 2D DRAWING	11
4.3 WRITTINGS ANYWHERE ON LAPTOP SCREEN	11
4.4 DRAWING ON PROJECTOR SCREEN	12
4.5 DETECTING DIFFERNT COLOR BALLS	12
4.6 DETECTING DIFFERNT COLORS	13
4.7 LEAF SHAPE DETECTION	13
6.1 LIVE SCREEN WINDOW	27
6.2 PAINT WINDOW	27
6.3 MASK WINDOW	28

CHAPTER 1 : INTRODUCTION

1.1 INTRODUCTION

If we want to draw our imagination by just waving our finger in the air we can do it with Air Canvas which can draw our imagination on live camera by just capturing the motion of a colored marker or any object with a camera. Here we are using a colored object at the tip of the finger as the marker or we can say that a color object to draw on screen.

We are going to use the one of computer vision techniques **OpenCV** to build this project. We are using Python as a programming language due to its exhaustive libraries and syntax is easy but understanding the basics it is also to implemented it in any OpenCV supported language.

Here we are mainly using Color Detection and tracking in order to achieve the objective. First color marker is detected and a mask is produced for detected marker. It also includes the further steps of morphological operations on the mask like Erosion and Dilation. Erosion is basically to reduces the impurities present in the mask and dilation is for further restores the eroded main mask.

1.2 PRESENT SYSTEM

ARTIFICIAL INTELLIGENCE

The term artificial intelligence was come in 1956 but now a days AI is getting more popular and this is because of increased data, advances and fast algorithms for process data, and good computing power and storage. At starting in 1950s-1970s first Neural Networks come then in 1980s-2010s Machine Learning come and now a day's Deep Learning with us.

As its name specifies we are giving artificially human intelligence to machine so that machine can also think and take decisions as a human can. The main goal of AI is include learning, problem-solving, perceptron.

Applications: Applications of AI are endless. Artificial Intelligence can be applied to different sectors, industries like – healthcare industries are dosing drugs for different surgical procedures. Some daily life application is also there like - Self-driving car, Object Recognition, Siri, Alexa, Netflix, Computer games in which human can play with computer etc.

COMPUTER VISION

In 1950s on Computer Vision some experiments took place, with the help of deep learning to detect the edges of objects to categories them like circles and squares. In 1970s first commercial use of CV take place as types or handwritten text with use of optical character recognition. This advancement was used for blind to recognize written text. When in the 1990s internet matured, then for analysis, facial recognition program making large sets of images available online. These increasing data sets helped for making it possible for machines to recognize and identify people in photos and videos.

In Artificial Intelligence there is a field called ‘Computer Vision’ that help computers to trains so that they can interpret and understand the visual world. Using Cameras and video we can capture digital images and with deep learning models machine can identify and classify objects accurately and then machine we react to it.

There are some numbers of factors have converged to bring improvements in computer vision:

- Mobile Technology with cameras
- Computing power
- Hardware designed
- New algorithms (like CNN)

With some of these advancements on computer vision the accuracy rate of object identification and classification increased from 50 percent to 99 percent. Today we have systems that are more accurate than human in quick detecting. Computer vision is a field of computer science which main is to focuses on replicating complex part of human vision system and it also enable computer to identify object.

1.3 PROPOSED SYSTEM

During this Covid situation everyone is study from online medium and for online studies a pen tab is one of the most important requirement or we can say that some simple and easy to use tool require. This can take place of a pen tab and it is easy to use. It is cheap version of pen tab and can full fill most of the requirements of a pen tab. This has some features like different colour pens and eraser that are also some important features.

Now days a pen tab is the most important requirement and this will help to provide a digital board that can operate using figure. A digital board will also be helpful in better understanding of students. We can draw in air using a simple Data Structure (Queue) for collect pixel and OpenCV for detect object. First we will read video and detect object then pixels on which object is moving collect in queue.

Through this we can provide many features in live camera like colour of different pen, eraser. Different features will also helpful for clear understanding of students and will make teaching easy for teachers. This will do a work similar to a pen tab or we can say that this will be a cheap version of pen tab. Pen tab is not effort able for every teacher so, will be a cheap version of that.

There are few steps to design this project:

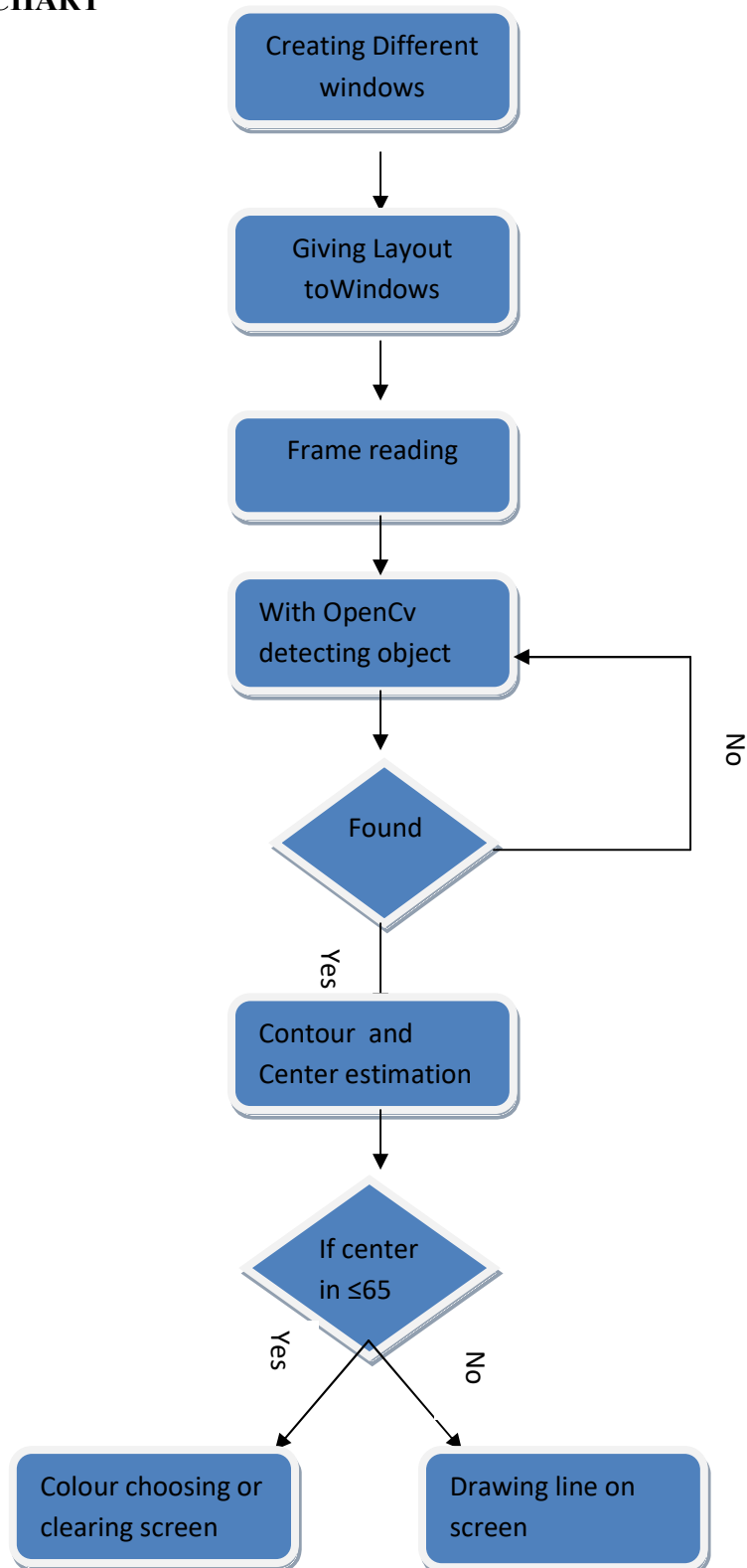
1. **Read frames in live camera** :- This can achieve with OpenCV module cv2
2. **Tracking by defining HSV values for color:** - We will find the HSV upper and lower range and will track color that came in that range.
3. **Create mask:** - We will a make for remove noise or impurities. This we can do with concept of Dilation and Erosion.
4. **Making contour:** - This will make a boundary around detected object.
5. **Tracking center of contour:** - By tracking center of contour we can find the exact pixels on which our contour in going.
6. **Storing pixel values in deque:** - We will store pixel values in deque and we will draw line by joining detected pixel with previous pixel.

We can also add some features like: - Eraser for erase for selected portion of screen, Choosing different color by selecting different colors, Scrolling of page to write in continue, and some feature like also able to write on anywhere on screen not only in selected regions on screen etc.

This system not require any data set to learn so, this is not require updates in data set with future change. And this system also not requires much future maintenance. We are simply using computer vision concept and basic logic to design this project so, with future we can keep this system as a base and can add features as per our requirements. It is very easy to understand because it is using simple logic and opencv concepts to recognize object so, any person that is new to Artificial Intelligence can also understand this project with some basic understanding of concepts that are using in this project.

CHAPTER 2 : SYSTEM DESIGN

2.1 FLOW CHART



Creating Different windows :-

In this project we are going to create four windows so, that we will get better understanding of project.

1. **Paint window :-** This window is used to make a window that will be similar to our paint screen and this is providing a better experience for using this project.
2. **Live Screen Window :-** This window is to detect object in live screen and through this window we will also be able to see that our object is in range of camera or not.
3. **Trackbar window:-** This window is used to adjust the trackbar values so that we can set color range according to our object color.
4. **Mask window :-** We are also designing a window to see that there is any impurity available with object or not. And with this we will also be able to see mask after erosion and dilation operations.

Giving Layout to Windows:-

In this we are giving different layout to our windows like in paint and live window we want some color rectangles and also want text in that rectangles.

Frame reading :-

In this we are reading frames with camera and this is happening with the help of OpenCv command. In OpenCV, video can be read frame either by using camera that is connected to a computer or by reading a video file. The first step for reading a video file is to create an object for videocapture. Videocapture argument can be either the device index or the name of the video file that we want to read.

In most of cases, we are using a camera connected to the system. So, we are passing '0' and OpenCV uses the only camera attached to the computer. When there are more than one camera connected to the computer then we can select the second camera by passing '1' as argument and third camera by passing '2' and so on.

Found:- This is basic condition checker where we are checking that object found or object detected or not and according to condition we are going to next step.

Contour and Center estimation :-

Drawing in three dimensions is not easy, but we can use contour map as a alternative for representing plots in 2D space. Contour map basically uses contours or color-coded regions to helps us to visualize 3D data in 2D. We can also use Contour maps to visualize the error surfaces in deep learning and machine learning to optimization techniques like Gradient descent, Momentum gradient descent etc.

In computer vision and image processing, there is image moments are often used to characterize the shape of an object in an image. These detected moments capture basic statistical properties of the shape or object, including the area of the object, the centroid (center (x, y) -coordinates of the object), orientation, along with other desirable properties.

If center in ≤ 65 :- This is a condition based on position of center and accordingly we are paerforming task.

Colour choosing or clearing screen :- If our center in area that is ≤ 65 then we storing pixel values according to chosen index values or we are clearing all dequees for clear all lines .

Drawing line on screen :- If out center is not in area of ≤ 65 then we are going to draw line according to selected color.

CHAPTER 3: HARDWARE AND SOFTWARE

3.1 HARDWARE REQUIREMENT

- We require a laptop with minimum 4 GB RAM and with a window 7 or greater Operating System so, that we will get a good processing speed in project processing.
- Laptop must have a camera facility so, that we can detect object in live camera.
- We also require a pen or any object with which we are comfortable to write on screen and this object should belongs to defined colour range in our project.

3.2 SOFTWARE REQUIREMENT

3.2.1 Anaconda :- Anaconda is a popular virtual environment because it have many of the tools that used in data science and machine learning with just one install of environment, so it's good for having short and easy setup. Like in Virtual environment, Anaconda is also using the concept of creating environments as per our requirement so it also provide different libraries and versions.

anaconda is a meta package that also includes all of the Python packages that comprising the Anaconda distribution. we can also install python 3.6 package and version in this new environment. There could be any package, like numpy 1.7 , or multiple packages can also possible.

3.2.2 Jupyter Notebook :-

Jupyter Notebook is an open-source application that allow us to create and share different documents that can also contain live code, visualizations and explanatory text. There use include : data cleaning and transformation, statistical modeling, machine learning and much more.

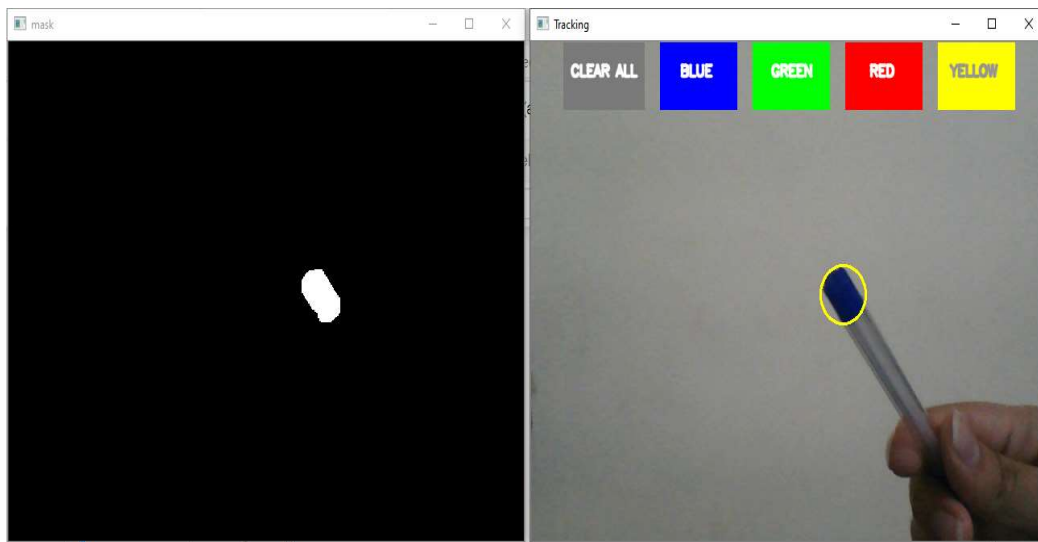
Jupyter Notebook also provide us an easy-to-use and interactive data science environment across different programming languages that does not only work as an IDE, but can also use as a presentation and education tool. It is a perfect tool for those who are just started with data science and machine learning.

CHAPTER 4: IMPLEMENTATION WORK DETAILS

4.1 REAL LIFE APPLICATIONS

4.1.1 AS A PEN TAB COPY

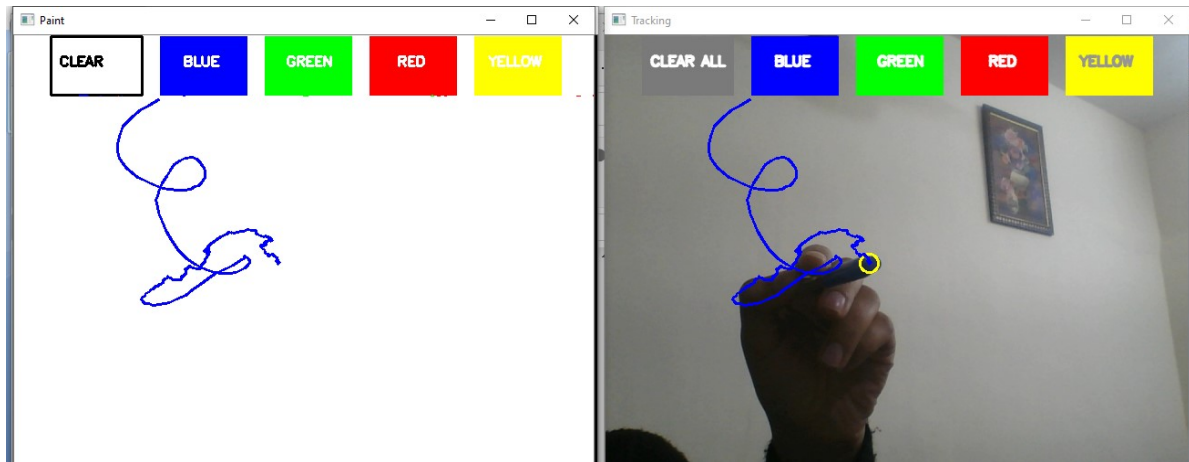
As we know that during this covid situation every school/college is running on a online mode and every teacher is teaching online and they require a digital pen to give better understanding to student but it is not effort able by every teacher in this situation, this project is providing a copy of pen tab that require only basic knowledge to operate this so, we can say that this is a cheaper version of pen tab.



4.1 PEN TAB COPY

4.1.2 FOR 2D DRAWING

In 3D drawing so many difficulties come and 3D is complex to operate. In 3D drawing we have think about angles and 3D drawing is not that much easy task to do. It is a simple version of 3D drawing where we can draw on 2D and our screen can give us a 2D surface to draw.



4.2 MAPPING 3D DRAWING IN 2D DRAWING

4.1.3 WRITTINGS ANYWHERE ON LAPTOP SCREEN

This can also use a base and with this we can also handle writing anywhere on screen like if some teacher is explaining some ppt or some study material that is in form of document or ppt then with advancement in this project they can also written and underline on that documents and ppt.



4.3 WRITTINGS ANYWHERE ON LAPTOP SCREEN

4.1.4 DRAWING ON PROJECTOR SCREEN

In future with improvement in facilities we can use this project in a very wide area and some of them we have discussed in above point. With time and requirement we can also improve this project with applications some of them like writing in projector screen and giving a experience of presentation.



4.4 DRAWING ON PROJECTOR SCREEN

4.1.5 DETECTING DIFFERNT COLOR BALLS

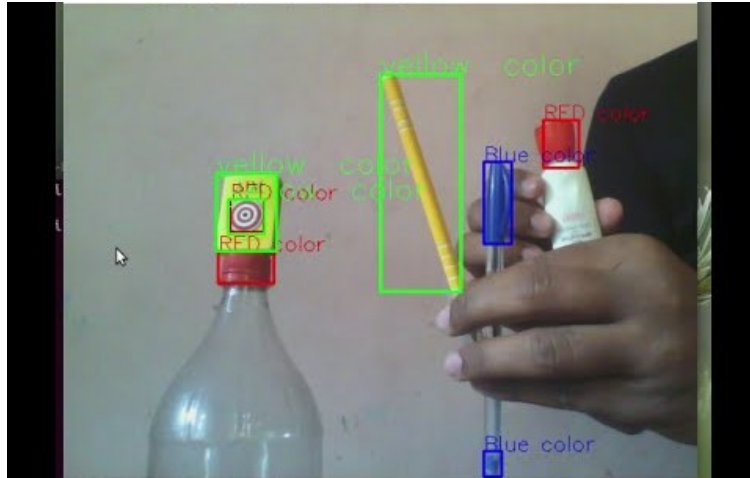
We can use colour detection in detection of different colour balls. And with the help of different color balls detection we can make group of same color balls.



4.5 DETECTING DIFFERNT COLOR BALLS

4.1.6 DETECTING DIFFERNT COLORS

We can detect different colors and with the help of this we can differentiate between object according to their colors. This can possible in a single live screen. Like in below screen there are different color objects and they all can differentiate with the help of their colors.



4.6 DETECTING DIFFERNT COLORS

4.1.7 LEAF SHAPE DETECTION

We can detect a leaf shape with color detection. We can detect different colour in a leaf and with the help of this we can detect a shape of leaf. There are different shades of colors available in a leaf and accordingly we can detect shape.



4.7 LEAF SHAPE DETECTION

4.2 DATA IMPLEMENTATION

4.2.1 LIBRARIES USED

We are importing some python libraries to use the chunk of inbuilt functions. And with the help of these libraries we will be able to use inbuilt functions and it remove the need of rewrite commonly used commands in our program. There are many libraries available in python but we are here using some of them like numpy, cv2, deque.

4.2.1.1 Numpy :-

NumPy stands for Numerical Python, it is a library which consisting multi dimensional array objects and for processing those arrays a collection of routines. With NumPy, different operations like mathematical, logical operations on different arrays can also be performed.

At a continuous place in computer memory unlike lists NumPy arrays stored, so processes easily can access, manipulate arrays very efficiently. In computer science this above behavior is known as locality of reference and it is the main reason that why NumPy is faster as compare to lists.

Numpy is also optimized and comfortable to work with available latest CPU architectures.

NumPy is a Python library which is written in Python, but most of the parts of numpy that require good and fast computation are written in C and C++.

Numpy is a fundamental package of Python for scientific computing. Numpy also contains many features including all these important ones:

- N-dimensional array object
- Different Sophisticated and broadcasting functions
- Providing tools for integration with C,C++ and Fortran code
- It is Fourier transform, Useful linear algebra, and also random number capabilities

4.2.1.2 OpenCV:-

OpenCV is a open-source library in Python, it is used for computer vision in, Machine Learning, Artificial intelligence, face recognition, etc. CV is an abbreviation of Opencv form a computer

vision, and it is defined as field of study which helps computers in understanding the different content of the digital images like photographs and videos.

The main purpose of computer vision is that to understand content of the different images. OpenCV extracts the available description from the pictures, that can be an object, or a text description, and it can be three-dimension model, etc.. Example, cars can also be facilitated by computer vision, that will be able to identify many different objects around the road and car, such as pedestrians, traffic signs, traffic lights, etc. and acts accordingly.

Computer vision give facility to computer to perform the same tasks as humans can with the same efficiency of human. There are mainly two main task that are defined below:

- **Object Classification** – In object classification, we first train our model on some dataset of particular objects, and then model classifies the new objects that belongs to one, more of training category.
- **Object Identification** - In object identification model identify some particular instance of object. Example – giving two faces in a single image and then tagging one as Virat Kohli ,other one as Rohit Sharma..

Using OpenCV we can develop computer vision real-time applications. It focuses on image processing and video capture. It's analysis include some features like face detection, object detection.

There are some OpenCV functions which we are going to use in our project these are as following:

1. **cv2.namedWindow:-** With the help of cv2.namedWindow() function we will be able to create different windows in our project for better understanding of processes and the parameter of this function shows the name of that new window.
2. **cv2.createTrackbar:-** With the help of cv2.createTrackbar() function we are creating trackbars and with the help of parameters we are able to define trackbar name, window name, min and max value for different trackbars, a default function.
3. **cv2.rectangle:-** This method is use draw a rectangle on image.

Syntax: *Cv2.rectangle (image, startpoint, endpoint, color, thickness_of_line)*

end_point :- This is a ending coordinates of out rectangle. Rectangle coordinates are represent using tuples that have two values (X and Y coordinate value).

color:- This is color of rectangle border line to be drawn.

4. **cv2.putText :-** It is a Python library to solve computer vision problems. This method is basically use to draw text string on image.

Syntax:*Cv2.putText(image_name, text, org, font_of_text, img_fontScale, color[, thick[, Type_of_line[, bottom_Left_Origin]]])*

Image_name: This is image on which we want text to be drawn.

text: This is text string that is going to be drawn.

org: This is coordinates of text string bottom-left corner in the image.

5. **Cv2.VideoCapture:-** This is a **video capture** object that is helpful in capture videos from webcam and after this we may perform desire operations on video.
6. **cv2.flip:-** This method use to flip a two dimensional array. This function flips a two dimensional array around horizontal, vertical, or can say on both axes.

Syntax: *cv2.flip(src, flip_Code[, dst])*

src: This is Input array.

dst: This is o/p array of same size, type as src.

flip_code: This is a flag to specify that how to flip array. Here 0 means flip around x axis with positive value example, 1 means flip around y axis. Negative value like example, -1 this mean flip around both x,y axes.

Return Value: This return image.

7. **cv2.cvtColor:-** This method use to convert image from one color space to another color space. This is very simple and we can also use this function, *cv2.cvtColor()*. Instead of pass image, we can just pass BGR values that we want.
8. **Cv2.getTrackbarPos:-** This method is used to find the trackabr postion that we given according to our object color.
9. **cv2.inRange:-** This Perform a basic thresholding operation by using OpenCV function . This Detect an object bases on range of pixel values that we have given in HSV color space.
10. **cv2.erode:-** This method use to perform operation like erosion on image. Basic idea of erosion just like soil erosion , it only erodes away boundaries of the foreground object and always keep trying to keep foreground in the white. This is normally performe on the binary image. This need two inputs, and one is for our original image, and second one call structuring element or can say kernel that decide nature of the operation.
11. **cv2.morphologyEx:-** This is also known as opening or can say that another name erosion follow by dilation. This is useful in remove the noise, and we have explained this above.
12. **cv2.dilate:-** This is opposite of erosion operation. Here pixel element is 1 if atleast any one pixel under kernel is 1. So, this is increases white region in image or foreground size in object increases. In case like erosion is followed by dilation, noise removal. Erosion removes white noise, but this is also shrink our object. So we use dilate in it. Since noise is not there, they would not come back, but our area of object increases. This is also useful in joining the broken parts of our object.
13. **cv2.findContours :-** This function is helpful in extract the contour from image. This works best in binary image, so we require first to apply threshold technique, and Sobel edges, etc.
14. **cv2.minEnclosingCircle :-** In this as per object area, parameter and centroid we are finding a minimum enclosing circle that will cover the object.
15. **cv2.circle :-** This method is useful in to draw circle on image.
16. **cv2.moments :-** With the help of this function we can found the center of min enclosing circle.
17. **cv2.line :-** This method is useful in to draw a line on the image.
18. **cv2.imshow :-** This method is useful in to show frames that we captures with the help of camera.

4.2.2 OBJECT DETECTION

OpenCV is a huge open-source python library this is for computer vision, and machine learning, and now it play a major role in real time operation that is very important in our today's system. With the help of using this, one of this can process image, videos for identify object, face, and hand writing of a human.

Object Detection is computer technology which is related to image processing, computer vision, and deep learning that deal with detection of objects in the image and video.

CHAPTER 5: SOURCE CODE

```
import numpy as np

import cv2

from collections import deque

#default called trackbar function
def setValues(x):

    print("")

# Creating the trackbars needed for adjusting the marker colour
cv2.namedWindow("Color detectors")

cv2.createTrackbar("Upper Hue", "Color detectors", 153, 180,setValues)
cv2.createTrackbar("Upper Saturation", "Color detectors", 255, 255,setValues)
cv2.createTrackbar("Upper Value", "Color detectors", 255, 255,setValues)
cv2.createTrackbar("Lower Hue", "Color detectors", 64, 180,setValues)
cv2.createTrackbar("Lower Saturation", "Color detectors", 72, 255,setValues)
cv2.createTrackbar("Lower Value", "Color detectors", 49, 255,setValues)

# Giving different arrays to handle colour points of different colour
bpoints = [deque(maxlen=1024)]
gpoints = [deque(maxlen=1024)]
```

```

rpoints = [deque(maxlen=1024)]
ypoints = [deque(maxlen=1024)]

# These indexes will be used to mark the points in particular arrays of specific colour
blue_index = 0
green_index = 0
red_index = 0
yellow_index = 0

#The kernel to be used for dilation purpose
kernel = np.ones((5,5),np.uint8)

colors = [(255, 0, 0), (0, 255, 0), (0, 0, 255), (0, 255, 255)]
colorIndex = 0

# Here is code for Canvas setup
paintWindow = np.zeros((471,636,3)) + 255
paintWindow = cv2.rectangle(paintWindow, (40,1), (140,65), (0,0,0), 2)
paintWindow = cv2.rectangle(paintWindow, (160,1), (255,65), colors[0], -1)
paintWindow = cv2.rectangle(paintWindow, (275,1), (370,65), colors[1], -1)
paintWindow = cv2.rectangle(paintWindow, (390,1), (485,65), colors[2], -1)
paintWindow = cv2.rectangle(paintWindow, (505,1), (600,65), colors[3], -1)

cv2.putText(paintWindow, "CLEAR", (49, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 0), 2, cv2.LINE_AA)

```

```

cv2.putText(paintWindow, "BLUE", (185, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255,
255, 255), 2, cv2.LINE_AA)

cv2.putText(paintWindow, "GREEN", (298, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255,
255, 255), 2, cv2.LINE_AA)

cv2.putText(paintWindow, "RED", (420, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255,
255, 255), 2, cv2.LINE_AA)

cv2.putText(paintWindow, "YELLOW", (520, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5,
(150,150,150), 2, cv2.LINE_AA)

cv2.namedWindow('Paint', cv2.WINDOW_AUTOSIZE)

```

```

# Loading the default webcam of PC.

```

```

cap = cv2.VideoCapture(0)

```

```

# Keep looping

```

```

while True:

```

```

    # Reading the frame from the camera

```

```

    ret, frame = cap.read()

```

```

    #Flipping the frame to see same side of yours

```

```

    frame = cv2.flip(frame, 1)

```

```

    hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

```

```

    u_hue = cv2.getTrackbarPos("Upper Hue", "Color detectors")

```

```

    u_saturation = cv2.getTrackbarPos("Upper Saturation", "Color detectors")

```

```

    u_value = cv2.getTrackbarPos("Upper Value", "Color detectors")

```

```

    l_hue = cv2.getTrackbarPos("Lower Hue", "Color detectors")

```

```

l_saturation = cv2.getTrackbarPos("Lower Saturation", "Color detectors")
l_value = cv2.getTrackbarPos("Lower Value", "Color detectors")
Upper_hsv = np.array([u_hue,u_saturation,u_value])
Lower_hsv = np.array([l_hue,l_saturation,l_value])


# Adding the colour buttons to the live frame for colour access
frame = cv2.rectangle(frame, (40,1), (140,65), (122,122,122), -1)
frame = cv2.rectangle(frame, (160,1), (255,65), colors[0], -1)
frame = cv2.rectangle(frame, (275,1), (370,65), colors[1], -1)
frame = cv2.rectangle(frame, (390,1), (485,65), colors[2], -1)
frame = cv2.rectangle(frame, (505,1), (600,65), colors[3], -1)

cv2.putText(frame, "CLEAR ALL", (49, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255,
255, 255), 2, cv2.LINE_AA)

cv2.putText(frame, "BLUE", (185, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255,
255), 2, cv2.LINE_AA)

cv2.putText(frame, "GREEN", (298, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255,
255), 2, cv2.LINE_AA)

cv2.putText(frame, "RED", (420, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255,
255), 2, cv2.LINE_AA)

cv2.putText(frame, "YELLOW", (520, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5,
(150,150,150), 2, cv2.LINE_AA)


# Identifying the pointer by making its mask
Mask = cv2.inRange(hsv, Lower_hsv, Upper_hsv)

Mask = cv2.erode(Mask, kernel, iterations=1)

```

```

Mask = cv2.morphologyEx(Mask, cv2.MORPH_OPEN, kernel)
Mask = cv2.dilate(Mask, kernel, iterations=1)

# Find contours for the pointer after identifying it
cnts,_ = cv2.findContours(Mask.copy(), cv2.RETR_EXTERNAL,
                           cv2.CHAIN_APPROX_SIMPLE)
center = None

# If the contours are formed
if len(cnts) > 0:
    # sorting the contours to find biggest
    cnt = sorted(cnts, key = cv2.contourArea, reverse = True)[0]
    # Get the radius of the enclosing circle around the found contour
    ((x, y), radius) = cv2.minEnclosingCircle(cnt)
    # Draw the circle around the contour
    cv2.circle(frame, (int(x), int(y)), int(radius), (0, 255, 255), 2)
    # Calculating the center of the detected contour
    M = cv2.moments(cnt)
    center = (int(M['m10'] / M['m00']), int(M['m01'] / M['m00']))

# Now checking if the user wants to click on any button above the screen
if center[1] <= 65:
    if 40 <= center[0] <= 140: # Clear Button
        bpoints = [deque(maxlen=512)]
        gpoints = [deque(maxlen=512)]

```

```

rpoints = [deque(maxlen=512)]
ypoints = [deque(maxlen=512)]

blue_index = 0
green_index = 0
red_index = 0
yellow_index = 0

paintWindow[67:,:,:] = 255
elif 160 <= center[0] <= 255:
    colorIndex = 0 # Blue
elif 275 <= center[0] <= 370:
    colorIndex = 1 # Green
elif 390 <= center[0] <= 485:
    colorIndex = 2 # Red
elif 505 <= center[0] <= 600:
    colorIndex = 3 # Yellow
else :
    if colorIndex == 0:
        bpoints[blue_index].appendleft(center)
    elif colorIndex == 1:
        gpoints[green_index].appendleft(center)
    elif colorIndex == 2:
        rpoints[red_index].appendleft(center)
    elif colorIndex == 3:

```

```

        ypoints[yellow_index].appendleft(center)

# Append the next dequeues when nothing is detected to avoid messing up
else:

    bpoints.append(deque(maxlen=512))

    blue_index += 1

    gpoints.append(deque(maxlen=512))

    green_index += 1

    rpoints.append(deque(maxlen=512))

    red_index += 1

    ypoints.append(deque(maxlen=512))

    yellow_index += 1


# Draw lines of all the colors on the canvas and frame
points = [bpoints, gpoints, rpoints, ypoints]

for i in range(len(points)):

    for j in range(len(points[i])):

        for k in range(1, len(points[i][j])):

            if points[i][j][k - 1] is None or points[i][j][k] is None:

                continue

            cv2.line(frame, points[i][j][k - 1], points[i][j][k], colors[i], 2)

            cv2.line(paintWindow, points[i][j][k - 1], points[i][j][k], colors[i], 2)


# Show all the windows
cv2.imshow("Tracking", frame)

cv2.imshow("Paint", paintWindow)

```

```
cv2.imshow("mask",Mask)
```

```
    # If the 'q' key is pressed then stop the application
```

```
    if cv2.waitKey(1) & 0xFF == ord("q"):
```

```
        break
```

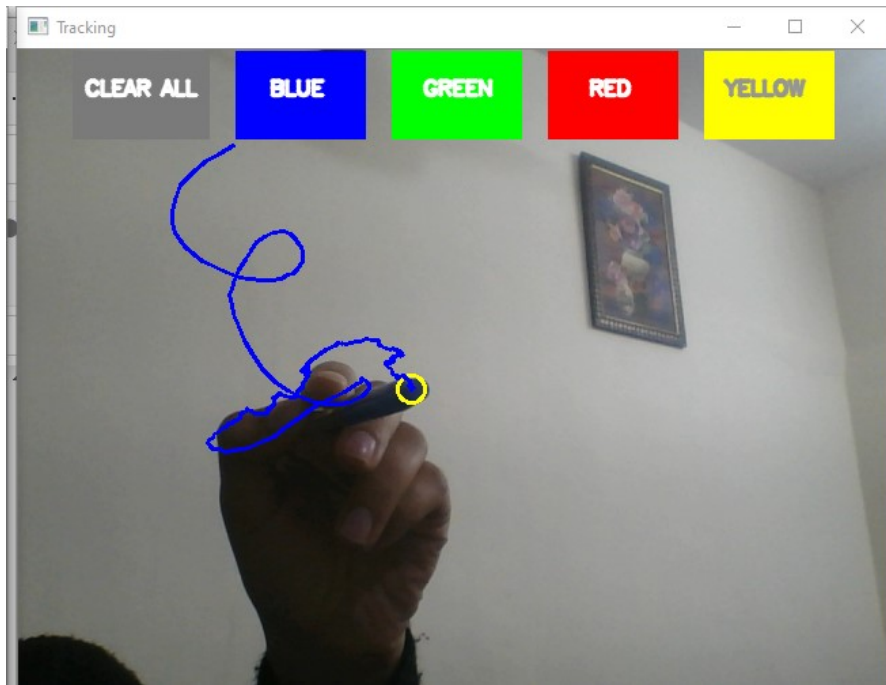
```
# Release the camera and all resources
```

```
cap.release()
```

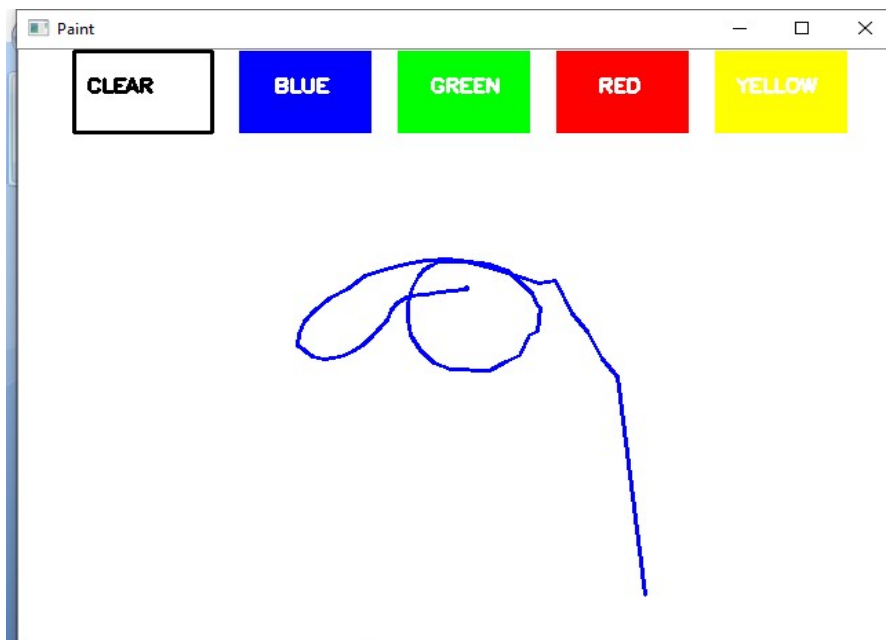
```
cv2.destroyAllWindows()
```


CHAPTER 6: INPUT/OUTPUT SCREENSHOTS

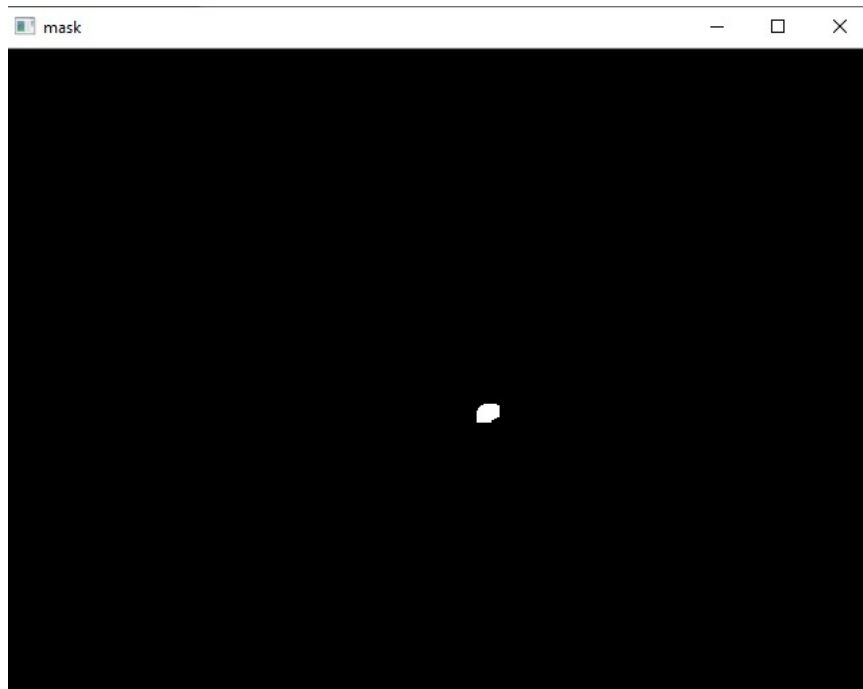
6.1 LIVE SCREEN WINDOW



6.2 PAINT WINDOW



6.3 MASK WINDOW



CHAPTER 7: CONCLUSION

7.1 LIMITATIONS

In future object detection, recognition has some massive potential across wide range of industry and this will keep increasing. Improving in transportation is effectiveness and yet a function and testing research because of criticality of transportation framework is being observed by frameworks.

Can't draw out of defined window:- Through this project we can draw on live screen only, we can't draw anywhere on screen. Through this we can draw on screens like paint window, and on live screen window but we can't draw on some other documents.

We are drawing in this by moving our figure on air or can say that on live screen by moving figure in air is not habitual thing. So, it require some time for human to take this in habit.

If we are drawing something on live screen and that screen is full with content then we can't scroll down page for write more in continuation. So, this is also a limitation of this project.

It will detect all object that will come in frame and matches with define HSV range. And if there is too many objects that are in color range of this project then it will detected all of them also.

7.2 SCOPE FOR FUTURE WORK

This can also use a base and with this we can also handle writing anywhere on screen like if some teacher is explaining some ppt or some study material that is in form of document or ppt then with advancement in this project they can also written and underline on that documents and ppt.

We can detect a leaf shape with color detection. We can detect different colour in a leaf and with the help of this we can detect a shape of leaf. There are different shades of colors available in a leaf and accordingly we can detect shape.

In future with improvement in facilities we can use this project in a very wide area and some of them we have discussed in above point. With time and requirement we can also improve this

project with applications some of them like writing in projector screen and giving a experience of presentation. We can detect different colors and with the help of this we can differentiate between object according to their colors. This can possible in a single live screen. Like in below screen there are different color objects and they all can differentiate with the help of their colors.

CHAPTER 8: BIBLIOGRAPHY

- Asst. Prof Ms.Vincy Joseph, Ms.Aditi Talpade, Ms. Nishitha Suvarna, Ms.Zeena Mendonca “Visual Gesture Recognition for Text writing in Air”, IEEE, 2018.
- Li Sun, Dimitrios Koutsonikolas, Souvik Sen, Kyu-Han Kim “WiDraw: Enabling Hands-free Drawing in the Air on Commodity WiFi Devices”, 2015.
- Keefe, D.F.; Zeleznik, R.C.; Laidlaw, D.H. “Drawing on Air: Input Techniques for Controlled 3D Line Illustration”, IEEE, 2007.





CHAPTER 9 ANNEXURES



Document Information

Analyzed document	Kishita Sureka-minor_report.docx (D90630008)
Submitted	12/27/2020 11:21:00 AM
Submitted by	Sunil Kumar Jangir
Submitter email	skjangir.set@modyuniversity.ac.in
Similarity	9%
Analysis address	skjangir.set.modyun@analysis.urkund.com

Sources included in the report

W	URL: https://github.com/varshneydevansh/object-detect-opencv/blob/master/README.md Fetched: 12/14/2019 5:02:26 PM	 3
W	URL: https://www.pyimagesearch.com/2016/02/01/opencv-center-of-contour/ Fetched: 12/27/2020 11:22:00 AM	 2
SA	Mody University of Science & Technology / Major Project Report Format-converted.pdf Document Major Project Report Format-converted.pdf (D76002947) Submitted by: kartiksharma.set@modyuniversity.ac.in Receiver: kartiksharma.set.modyun@analysis.urkund.com	 2
W	URL: https://www.javatpoint.com/opencv Fetched: 12/13/2019 6:36:10 PM	 4
W	URL: http://la-ferme-du-chateau.be/secret-stars-kjema/put-text-on-image-opencv.html Fetched: 12/22/2020 8:09:35 PM	 1