**MAULANA AZAD**

**NATIONAL INSTITUTE OF TECHNOLOGY BHOPAL INDIA, 462003**



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**AirBoard Painter**

**Minor Project Report Semester VI**

#### Submitted by:

Shashank Singh 181112060

Ankit Kumar 181112086

Vikas Sao 181112051

Kanishk Singh Pujari 181112076

**Under the Guidance of**

## Dr. Mansi Gyanchandani

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**Session: 2020-2021**

MAULANA AZAD

NATIONAL INSTITUTE OF TECHNOLOGY BHOPAL INDIA, 462003



### DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

# CERTIFICATE

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

This is to certify that the project report carried out on “AirBoard Painter ” by

the 3'd year students:

Shashank Singh 181112060

Ankit Kumar 181112086

Vikas Sao 181112051

Kanishk Singh Pujari 181112076

Have successfully completed their project in partial fulfilment of their Degree in Bachelor of

Technology in Computer Science and Engineering.

Dr . Mansi Gyanchandani

( Minor Project mentor )

**DECLARATION**

We, hereby declare that the following report which is being presented in the Minor Project Documentation Entitled as “ AirBoard Painter ” is an authentic documentation of our own original work under the guidance of Dr . Mansi Gyandchandani , Associate Professor, Department Of Computer Science and Engineering, MANIT Bhopal. The following project and its report, in part or whole, has not been presented or submitted by us for any purpose in any other institute or organization. Any contribution made to the research by others, with whom we have worked at Maulana Azad National Institute of Technology, Bhopal or elsewhere, is explicitly acknowledged in the report.

Shashank Singh 181112060

Ankit Kumar 181112086

Vikas Sao 181112051

Kanishk Singh Pujari 181112076

**ACKNOWLEDGEMENT**

With due respect, we express our deep sense of gratitude to our respected guide and coordinator Dr . Mansi Gyandchandani, for his valuable help and guidance. We are thankful for the encouragement that he has given us in completing this project successfully.

It is imperative for us to mention the fact that the report of minor project could not have been accomplished without the periodic suggestions and advice of our project guide Dr . Mansi Gyandchandani and project coordinators Dr. Dhirendra Pratap Singh and Dr. Jaytrilok Choudhary.

We are also grateful to our respected director Dr. N. S. Raghuwanshi for permitting us to utilize all the necessary facilities of the college.

We are also thankful to all the other faculty, staff members and laboratory attendants of our department for their kind cooperation and help. Last but certainly not the least; we would like to express our deep appreciation towards our family members and batch mates for providing the much needed support and encouragement.

**ABSTRACT**

Exploring the possibility of computer vision is both interesting from a scientific point of view and something that could be beneficial to virtual graphics industry .

We will be using the computer vision techniques of **OpenCV** to build this project. The preferred language is Python due to its exhaustive libraries and easy to use syntax but understanding the basics it can be implemented in any OpenCV supported language.

Here Color Detection and tracking are used in order to achieve the objective. The color marker is detected and a mask is produced. It includes the further steps of morphological operations on the mask produced which are Erosion and Dilation. Erosion reduces the impurities present in the mask and dilation further restores the eroded main mask.

To perform video tracking, an algorithm analyzes sequential video frames and outputs the movement of targets between the frames. There are a variety of algorithms, each having strengths and weaknesses. Considering the intended use is important when choosing which algorithm to use. There are two major components of a visual tracking system: target representation and localization, as well as filtering and data association.

Video tracking is the process of locating a moving object (or multiple objects) over time using a camera. It has a variety of uses, some of which are: human-computer interaction, security and surveillance, video communication and compression, augmented reality, traffic control, medical imaging and video editing

#### Keywords: OpenCV , DIP , Color Detection , Contours etc .

**TABLE OF CONTENTS**

1. [Introduction 1](#_TOC_250002)
2. [Literature review and survey ......2](#_TOC_250001)
3. [Proposed Work .3](#_TOC_250000)
   1. Data set
   2. Image Features
      1. Drawing on Canvas
      2. Contour Matching
4. **Methodology .......**6

Working with Canvas

Color Tracking

Color Detection

Storing the Shapes

Drawing on Canvas

1. **Tools and technology used (hardware and software) 9**
   1. Software requirement
   2. Hardware requirement
2. **Implementation 11**
3. **Screenshots and results … 22**
4. **Conclusion… 23**
5. **Future Scope… 23**
6. **References 23**
7. **Introduction**

Computer vision is a huge part of the data science/AI domain. Sometimes, computer vision engineers have to deal with videos. Here, we aim to shed light on video processing — using [**Python**](https://opendatascience.com/category/tools-languages/python/), of course.

This might be obvious for some, but nevertheless, video streaming is not a continuous process, but a discrete one.

That means, each time we deal with videos, we are actually dealing with the sequence of frames themselves. Each frame is just an image, which might be represented as an m x n array of pixels, where (m,n) is picture size. Each pixel might be represented as color intensity, depending on which color model we are using (gray-scale, RGB, or even multispectrum).

Often, we have to capture live stream with a camera. OpenCV provides a very simple interface to do this. Let's capture a video from the camera (I am using the built-in webcam on my laptop), convert it into grayscale video and display it. Just a simple task to get started.

To capture a video, you need to create a **VideoCapture** object. Its argument can be either the device index or the name of a video file. A device index is just the number to specify which camera. Normally one camera will be connected (as in my case). So I simply pass 0 (or -1). You can select the second camera by passing 1 and so on. After that, you can capture frame-by-frame. But at the end, don't forget to release the capture.

1. **Literature review and Survey**

Due to object detection’s close relationship with video analysis and image understanding, it has attracted much research attention in recent years. Traditional object detection methods are built on handcrafted features and shallow trainable architectures. Their performance easily stagnates by constructing complex ensembles which combine multiple low-level image features with high-level context from object detectors and scene classifiers. With the rapid development in deep learning, more powerful tools, which are able to learn semantic, high-level, deeper features, are introduced to address the problems existing in traditional architectures. As distinct specific detection tasks exhibit different characteristics, we also briefly survey several specific tasks, including salient object detection, face detection and pedestrian detection. Experimental analyses are also provided to compare various methods and draw some meaningful conclusions. Finally, several promising directions and tasks are provided to serve as guidelines for future work in both object detection and relevant neural network based learning systems.

Bhumika Gupta (2017) et a;. proposed object detection is a well-known computer technology connected with computer vision and image processing that focuses on detecting objects or its instances of a certain class (such as humans, flowers, animals) in digital images and videos. There are various applications of object detection that have been well researched including face detection, character recognition, and vehicle calculator. Object detection can be used for various purposes including retrieval and surveillance. In this study, various basic concepts used in object detection while making use of OpenCV library of python 2.7, improving the efficiency and accuracy of object detection are presented.

Kartik Umesh Sharma (2017) et al, proposed an object detection system finds objects of the real world present either in a digital image or a video, where the object can belong to any class of objects namely humans, cars, etc. In order to detect an object in an image or a video the system needs to have a few components in order to complete the task of detecting an object, they are a model database, a feature detector, a hypothesiser and a hypothesiser verifier. This paper presents a review of the various techniques that are used to detect an object, localise an object, categorise an object, extract features, appearance information, and many more, in images and videos. The comments are drawn based on the studied literature and key issues are also identified relevant to the object detection. Information about the source codes and online datasets is provided to facilitate the new researcher in object detection area. An idea about the possible solution for the multi class object detection is also presented. This paper is suitable for the researchers who are the beginners in this domain.

1. **Proposed Work**

**Dataset:**

Colors are made up of 3 primary colors; red, green, and blue. In computers, we define each color value within a range of 0 to 255. So in how many ways we can define a color? The answer is 256\*256\*256 = 16,581,375. There are approximately 16.5 million different ways to represent a color. In our dataset, we need to map each color’s values with their corresponding names. But don’t worry, we don’t need to map all the values. We will be using a dataset that contains RGB values with their corresponding names. The CSV file for our dataset has been taken from this link:



**Extracting data from CSV files:**

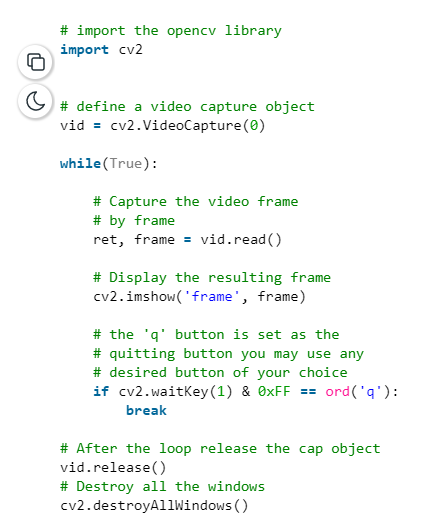
The pandas library is very useful when we need to perform various operations on data files like CSV. **pd.read\_csv()** reads the CSV file and loads it into the pandas DataFrame. We have assigned each column with a name for easy accessing.

**Image Features:**

Image feature is a simple image pattern, based on which we can describe what we see on the image. For example cat eye will be a feature on a image of a cat. The main role of features in computer vision(and not only) is to transform visual information into the vector space. This give us possibility to perform mathematical operations on them, for example finding similar vector(which lead us to similar image or object on the image)



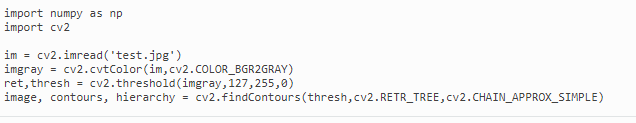
* 1. **Drawing running images from video OpenCV:**
* Use cv2.VideoCapture() to get a video capture object for the camera.
* Set up an infinite while loop and use the read() method to read the frames using the above created object.
* Use cv2.imshow() method to show the frames in the video.
* Breaks the loop when the user clicks a specific key.



* 1. **Contour Matching:**

Contours can be explained simply as a curve joining all the continuous points (along the boundary), having same color or intensity. The contours are a useful tool for shape analysis and object detection and recognition.

* For better accuracy, use binary images. So before finding contours, apply threshold or canny edge detection.
* findContours function modifies the source image. So if you want source image even after finding contours, already store it to some other variables.
* In OpenCV, finding contours is like finding white object from black background. So remember, object to be found should be white and background should be black.
* See, there are three arguments in **cv2.findContours()** function, first one is source image, second is contour retrieval mode, third is contour approximation method. And it outputs the image, contours and hierarchy. contours is a Python list of all the contours in the image. Each individual contour is a Numpy array of (x,y) coordinates of boundary points of the object.



1. **Methodology**

### **How Air Canvas type of Computer Vision Projects Work ?**

Here, We will see the working of this computer vision project in four major points :

1. Understanding the HSV (Hue, Saturation, Value) color space for **Color Tracking.**And tracking the small colored object at finger tip.
2. Detecting the Position of Colored object at finger top and forming a circle over it. That is [**Contour**](https://docs.opencv.org/3.4/d4/d73/tutorial_py_contours_begin.html)**Detection.**
3. Tracking the fingertip and drawing points at each position for air canvas effect. That is **Frame Processing.**
4. Fixing the Minor Details of the code to function the program smoothly. **Algorithmic Optimization.**

#### Colour Tracking of Object at fingertip.

Firts of all, The incoming image from the webcam is to be converted to the [HSV colour space](https://en.wikipedia.org/wiki/HSL_and_HSV) for **detecting the colored object at the tip of finger.** The below code snippet converts the incoming image to the HSV space, which is **very suitable and perfect color space for Color tracking.**

Now, We will make the[Trackbars](https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_gui/py_trackbar/py_trackbar.html) to arrange the HSV values to the required range of color of the colored object that we have placed at our finger. The various HUE and other ranges of different colors can be seen [here.](http://www.workwithcolor.com/blue-color-hue-range-01.htm)

Now, When the trackbars are setup, we will get the realtime value from the trackbars and create range. This range is a [numpy](https://numpy.org/) structure which is used to be passed in the function[cv2.inrange( ).](https://docs.opencv.org/3.4/da/d97/tutorial_threshold_inRange.html)This function returns the Mask on the colored object. This Mask is a black and white image with white pixels at the position of the desired color.

#### Contour Detection of the Mask of Color Object

Now, After detecting the Mask in Air Canvas, Now is the time to locate its center position for drawing the Line. Here, In the below Snippet of Code, We are performing some morphological operations on the Mask, to make it free of impurities and to detect contour easily.

#### Drawing the Line using the position of Contour

Now Comes the real logic behind this Computer Vision project, We will form a [python deque](https://docs.python.org/3/library/collections.html) (A data Structure). The deque will store the position of the contour on each successive frame and we will use these stored points to make a line using OpenCV drawing functions.

Firstly Make Four deques, for four distinct colors of the project :

Now, we will use the position of the contour to make decision, if we want to click on a button or we want to draw on the sheet. We have arranged some of the buttons on the top of Canvas, if the pointer comes into their area, we will trigger their method. We have four buttons on the canvas, drawn using OpenCV.

* Clear : Which clears the screen by emptying the deques.
* Red : Changes the marker to red color using color array.
* Green : Changes the marker to Green color using color array.
* Yellow : Changes the marker to Yellow color using color array.
* Blue : Changes the marker to Blue color using color array.

Also, to avoid drawing when contour is not present, We will Put a else condition which will capture that instant.

#### Drawing the points

Now we will draw all the points on the positions stored in the deques, with respective colour.

#### Features of the Air canvas

1. Can track any specific colored pointer .
2. User can draw in four different colors and even change them without any hussle.
3. Able to rub the board with a single location at the top of the screen.
4. No need to touch the computer once the program is run.
5. **Tools and technologies used**
   1. **Software requirement**

The various tools and technologies to be used are as follows:

I) Python Libraries to implement Machine Learning Models -

* **Pandas** - pandas is a software library written for the Python programming language for data manipulation and analysis.
* **NumPy** - NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.
* **Scikit learn** - Scikit is an open source Python library that implements a range of machine learning, pre-processing, cross-validation and visualization algorithms using a unified interface.
* **OpenCV -** OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.
* To build the machine learning models we used The **Jupyter Notebook**. **Jupyter**

**Notebook** is an open-source web application that can be used to implement statistical modelling, data visualization, machine learning etc.

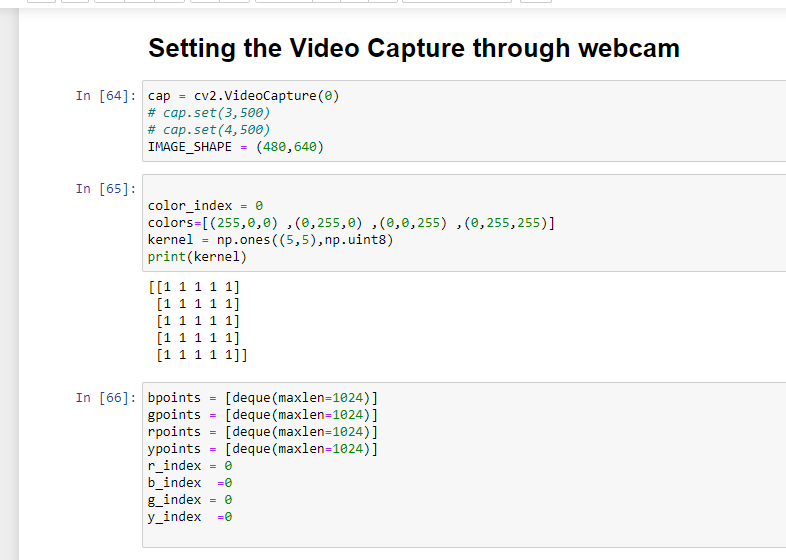
* **Collections -** The collection Module in Python provides different types of containers. A Container is an object that is used to store different objects and provide a way to access the contained objects and iterate over them. Some of the built-in containers are [Tuple](https://www.geeksforgeeks.org/python-tuples/), [List](https://www.geeksforgeeks.org/python-list/), [Dictionary](https://www.geeksforgeeks.org/python-dictionary/), etc. In this article, we will discuss the different containers provided by the collections module.
* **Matplotlib - Matplotlib** is a [plotting](https://en.wikipedia.org/wiki/Plotter) [library](https://en.wikipedia.org/wiki/Library_(computer_science)) for the [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) programming language and its numerical mathematics extension [NumPy](https://en.wikipedia.org/wiki/NumPy). It provides an [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) [API](https://en.wikipedia.org/wiki/API) for embedding plots into applications using general-purpose [GUI toolkits](https://en.wikipedia.org/wiki/GUI_toolkit) like [Tkinter](https://en.wikipedia.org/wiki/Tkinter), [wxPython](https://en.wikipedia.org/wiki/WxPython), [Qt](https://en.wikipedia.org/wiki/Qt_(software)), or [GTK](https://en.wikipedia.org/wiki/GTK).
  1. **Hardware requirement**

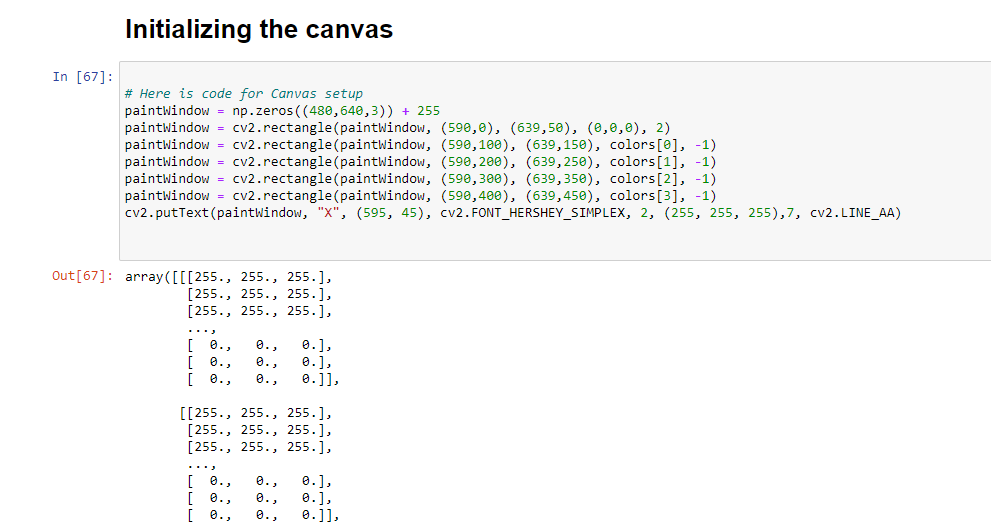
Working Computer system

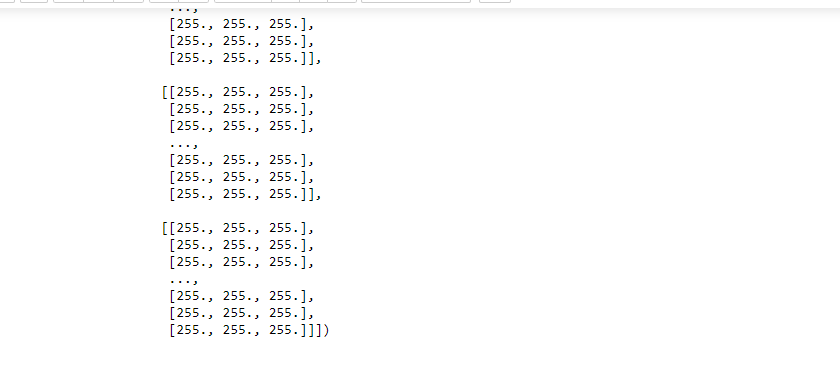
Working WebCam System

Processor Requirement: Intel I3 core and above. Memory Requirement: 4GB and above.

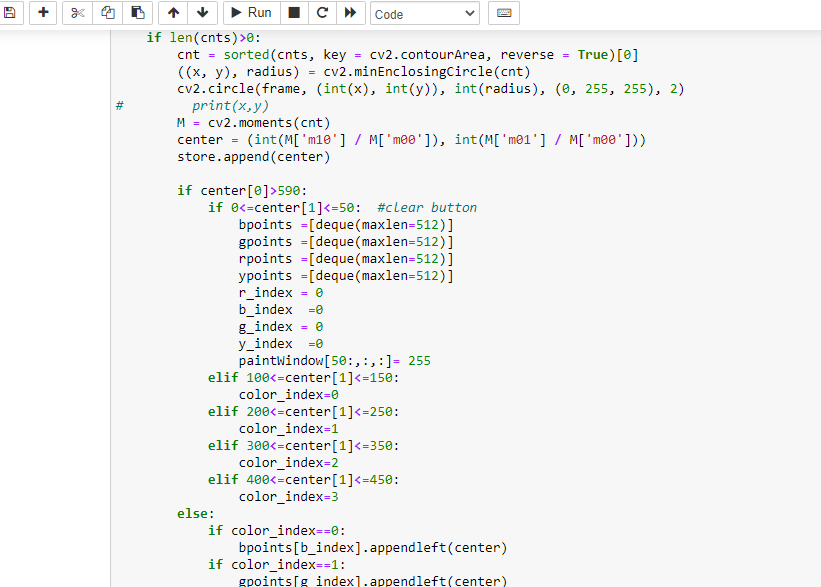
1. **Implementation & Code**
   1. **Code Snippets:**

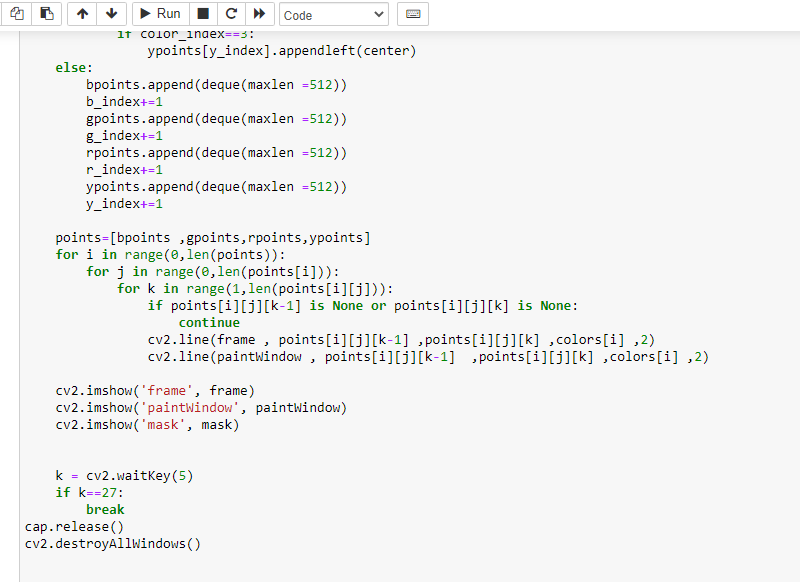
 

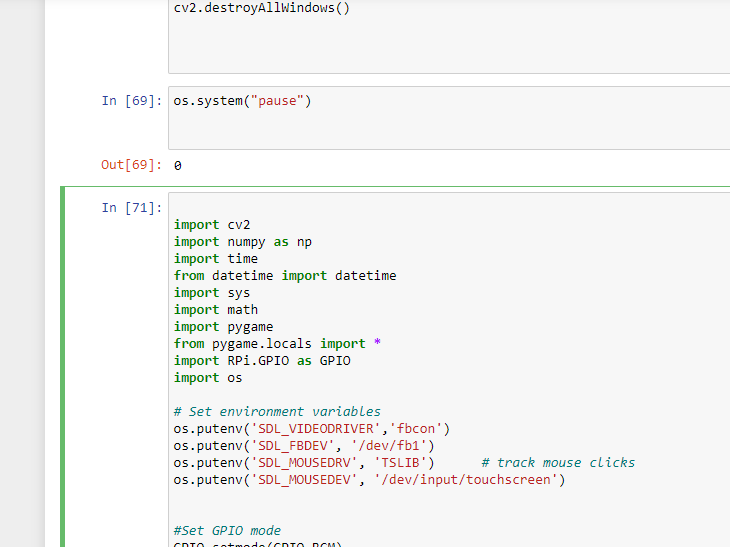


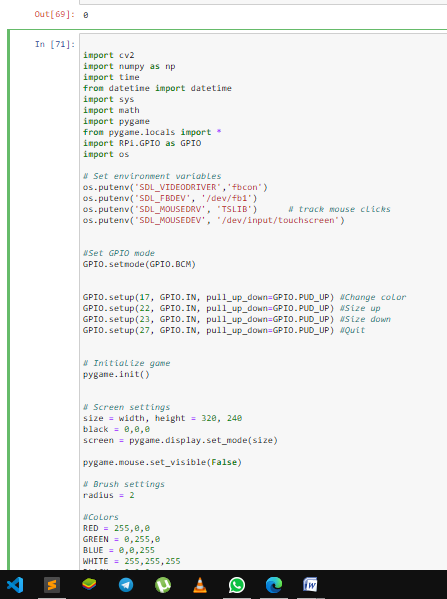


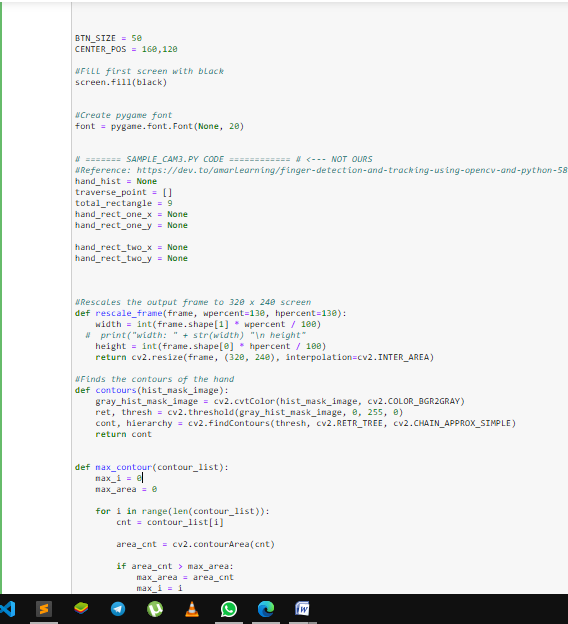


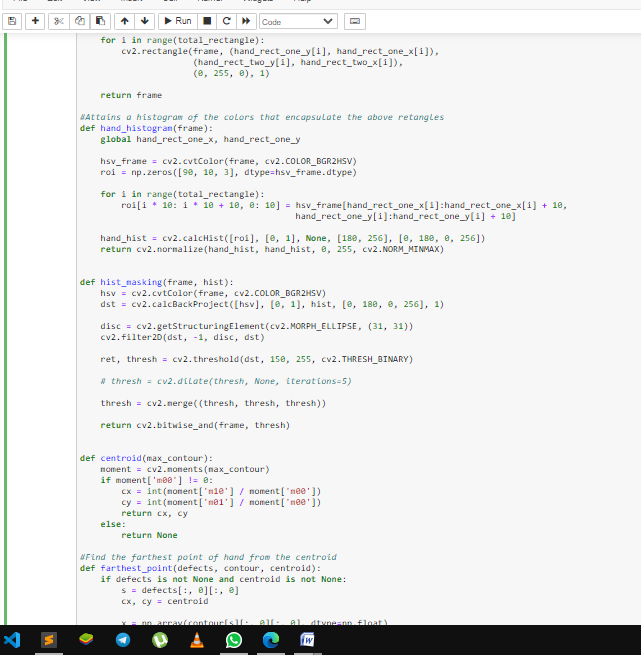


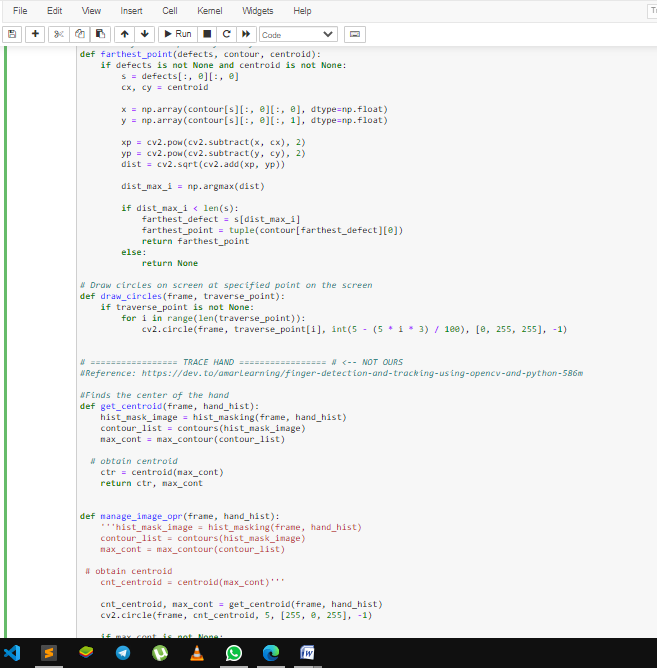


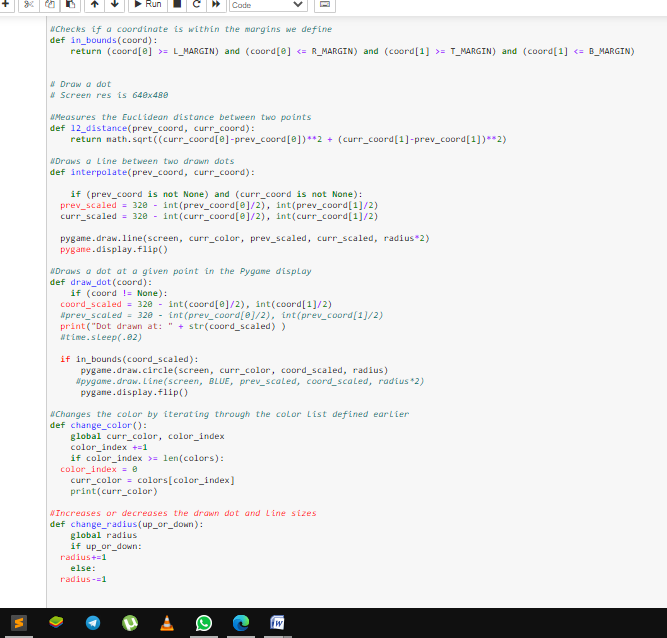


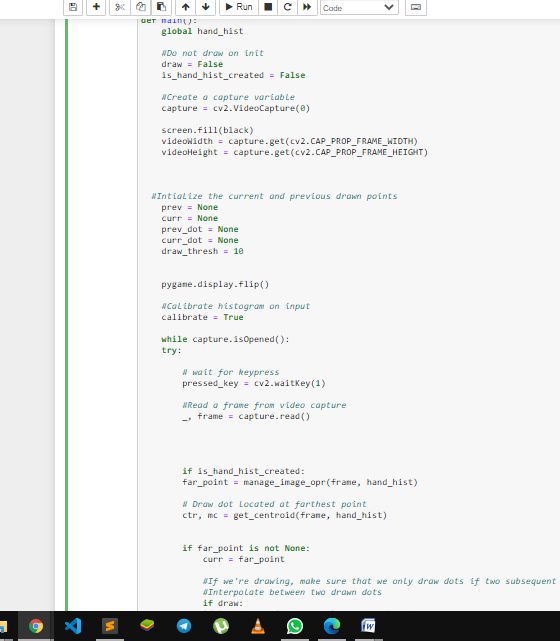


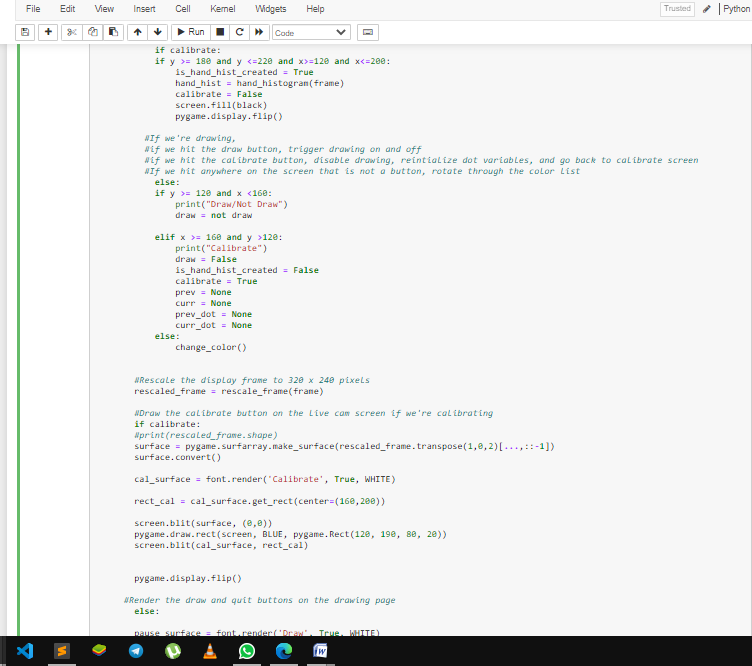


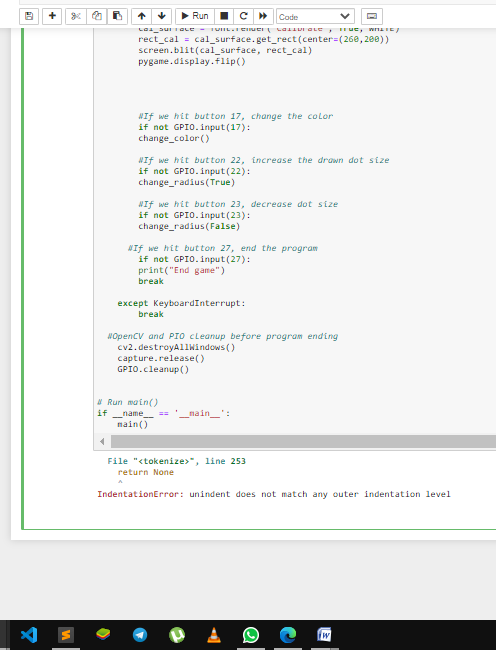




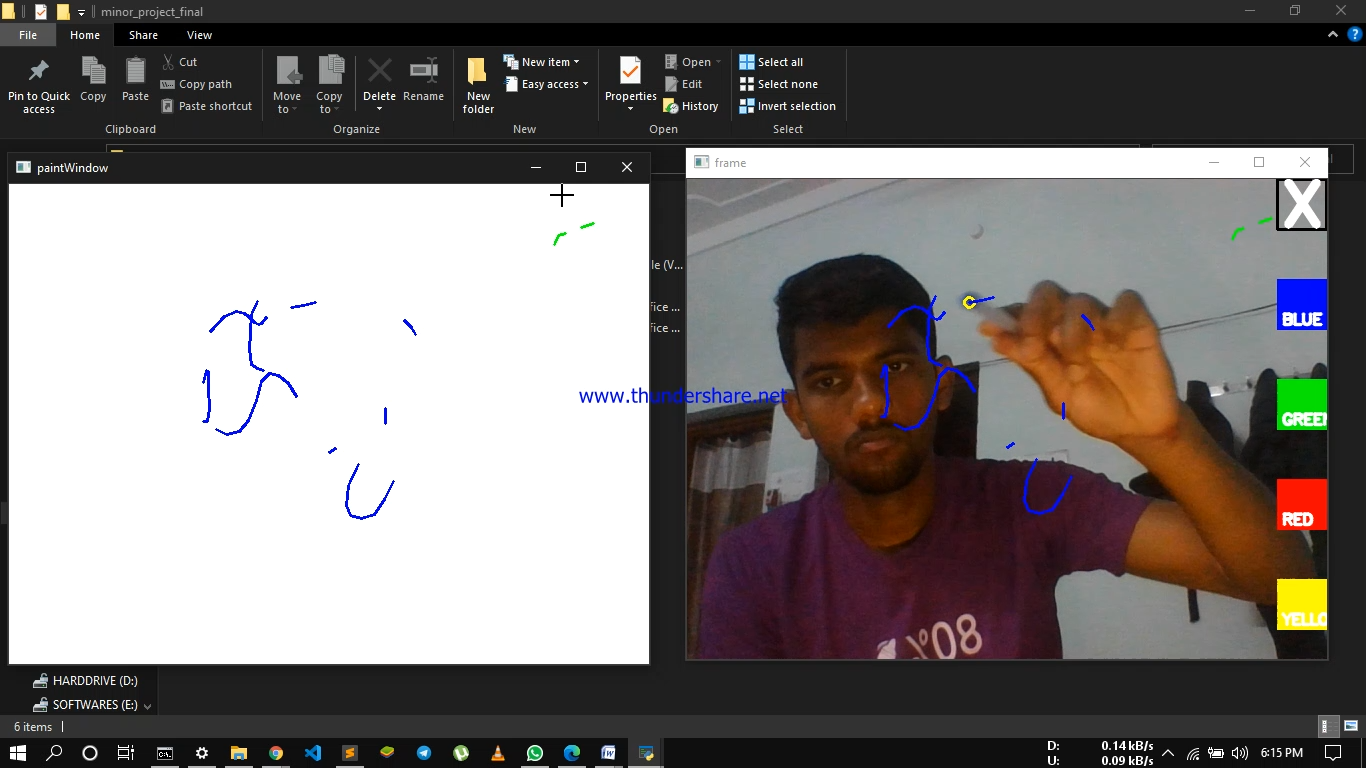


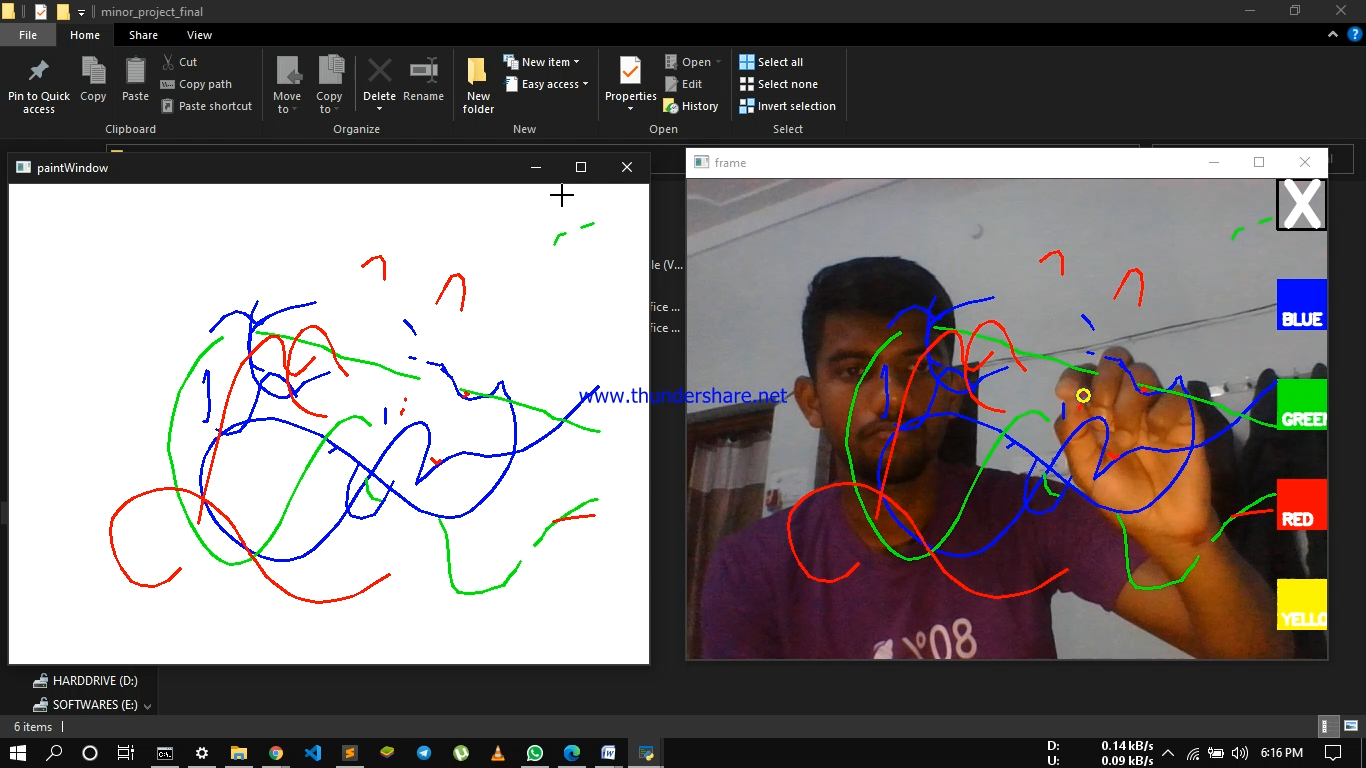


****

****

* 1. **Screenshots after running:**





1. **Conclusion**

This project is relevant to Virtual Developers and Artificial Intelligence. OpenCV is at work in robotics systems—picking let‐ tuce, recognizing items on conveyor belts, helping self-driving cars see, flying quadrotors, doing tracking and mapping in virtual and augmented reality systems, helping unload trucks and pallets in distribution centers, and more—and is built into the Robotics Operating System (ROS). It is used in applications that promote mine safety, prevent swimming pool drownings, process Google Maps and streetview imagery, and implement Google X robotics .

1. **Future Scope**

OpenCV is the most popular open source computer vision library in the world. It implements over 2500 optimized algorithms, works on all major operating systems, is available in multiple languages and is free for commercial use. This talk will primarily provide a technical update on OpenCV: What’s new in OpenCV 4.0? What is the Graph API? Why are we so excited about the Deep Neural Network (DNN) module in OpenCV? (Short answer: It is one of the fastest inference engines on the CPU.) We will also share plans for the future of OpenCV, including new algorithms that we plan to add through the Google Summer of Code this year. We will also briefly share info on the new Open Source Vision Foundation (OSVF), and on OpenCV’s sister organizations, CARLA and Open3D, and some of the initiatives planned by these organizations.

1. **References**
2. Learning OpenCV - [Gary Bradski Adrian Kaehler](https://www.amazon.com/_/dp/0596516134?tag=oreilly20-20)
3. Implementation - <http://infoaryan.com/>
4. [www.towardsdatascience.com](http://www.towardsdatascience.com/)
5. [www.medium.com](http://www.medium.com/)
6. Spotipy python documentation: <https://spotipy.readthedocs.io/en/2.13.0/>
7. OpenCV - <https://opencv.org/>
8. <https://scikit-learn.org/stable/>Scikit learn documentation Python
9. Face Detection & Recognition Report 08/09/2017 Andreea Diana Dinca, IT Innovation Intern ,

<mailto:itinnovation@contacts.bham.ac.uk>