**Project Title : Draw in air using Video Tracking and Finding Contour of Interest**

A Project Report Submitted for the partial fulfilment of Degree in

MASTER OF COMPUTER APPLICATIONS

by

Anjali Tiwari (1900520140007)

Pranshi Gaur (1900520140040)

Prafull Panday (1900520140038)

Subhadra Yadav (1900520140056)

*Under the supervision of*

Er AJIT SHUKLA

**

Department of Computer Science and Engineering

**Institute of Engineering and Technology**

**Dr. A.P.J. Abdul Kalam Technical University, Lucknow, Uttar Pradesh**

**July,2021**

**Contents**

DECLARATION ..................................................................................................

i. CERTIFICATE .................................................................................................

ii. ACKNOWLEDGEMENT................................................................................

1. INTRODUCTION..........................................................................................

2. LITERATURE REVIEW

3. METHODOLOGY

3.1

3.1.1

3.1.2

3.2

3.2.1

3.2.2

4. EXPERIMENTAL RESULTS

5. CONCLUSIONS

vi. REFERENCES

**Declaration**

We hereby declare that this submission is our own work and that, to the best of our belief and knowledge, it contains no material previously published or written by another person or material which to a substantial error has been accepted for the award of any degree or diploma of university or other institute of higher learning, except where the acknowledgement has been made in the text. The project has not been submitted by us at any other institute for requirement of any other degree.

Submitted by: - Date : 2nd August 2021

Anjali Tiwari- (1900520140007)

Pranshi Gaur - (1900520140040)

Prafull Panday- (1900520140038)

Subhadra Yadav- (1900520140056)

**Certificate**

This is to certify that the project report entitled “Draw in air using Video Tracking and Finding Contour of Interest” presented by Anjali Tiwari, Pranshi Gaur, Prafull Panday, Subhadra Yadav in the partial fulfilment for the award of Master of Computer Application, is a record of work carried out by them under my supervision and guidance at the Department of Computer Science and Engineering at Institute of Engineering and Technology, Lucknow.

It is also certified that this project has not been submitted at any other Institute for the award of any other degrees to the best of my knowledge.

Er Ajit Shukla

Department of Computer Science and Engineering

Institute of Engineering and Technology, Lucknow

**Chapter 1**

**Introduction**

Webcam paint is painting using an object (in our case it’s pen) to track movement of an object-of-interest using which a user can draw on the screen by moving the object around. This technology uses OpenCV with Python to perform object identification and tracking.

Abstract on Computer Vision and OpenCV- Computer vision is an interdisciplinary field that deals with how computers can be made to gain high-level understanding from digital images or videos. From the perspective of engineering, it seeks to automate tasks that the human visual system can do. "Computer vision is concerned with the automatic extraction, analysis and understanding of useful information from a single image or a sequence of images. It involves the development of theoretical and algorithmic basis to achieve automatic visual understanding." As a scientific discipline, computer vision is concerned with the theory behind artificial systems that extract information from images. The image data can take many forms, such as video sequences, views from multiple cameras, or multidimensional data from a medical scanner. As a technological discipline, computer vision seeks to apply its theories and models for the construction of computer vision systems. OpenCV (Open-Source Computer Vision), a cross- platform and free to use library of functions is based on real time Computer Vision which supports Deep Learning frameworks that aids in image and video processing. In Computer Vision, the principal element is to extract the pixels from the image so as to study the objects and thus understand what it contains. Below are a few key aspects that Computer Vision seeks to recognize in the photographs:

Object Detection: The location of the object.

Object Recognition: The objects in the image, and their positions.

Object Classification: The broad category that the object lies in.

Object Segmentation: The pixels belonging to that object

History of OpenCV- Officially launched in 1999 the OpenCV project was initially an [Intel Research](https://en.wikipedia.org/wiki/Intel_Research_Lablets) initiative to advance [CPU](https://en.wikipedia.org/wiki/Central_processing_unit)-intensive applications, part of a series of projects including [real-time](https://en.wikipedia.org/wiki/Real-time_computing) [ray tracing](https://en.wikipedia.org/wiki/Ray_tracing_(graphics)) and [3D display](https://en.wikipedia.org/wiki/3D_Display) walls. The main contributors to the project included a number of optimization experts in Intel Russia, as well as Intel's Performance Library Team. In the early days of OpenCV, the goals of the project were described as:

* Advance vision research by providing not only open but also [optimized code](https://en.wikipedia.org/wiki/Code_optimization) for basic vision infrastructure. No more [reinventing the wheel](https://en.wikipedia.org/wiki/Reinventing_the_wheel).
* Disseminate vision knowledge by providing a common infrastructure that developers could build on, so that code would be more readily readable and transferable.
* Advance vision-based commercial applications by making [portable](https://en.wikipedia.org/wiki/Portability_(computer_science)), performance-optimized code available for free – with a license that did not require code to be open or free itself.

The first alpha version of OpenCV was released to the public at the [IEEE Conference on Computer Vision and Pattern Recognition](https://en.wikipedia.org/wiki/Conference_on_Computer_Vision_and_Pattern_Recognition) in 2000, and five betas were released between 2001 and 2005. The first 1.0 version was released in 2006. A version 1.1 "pre-release" was released in October 2008.

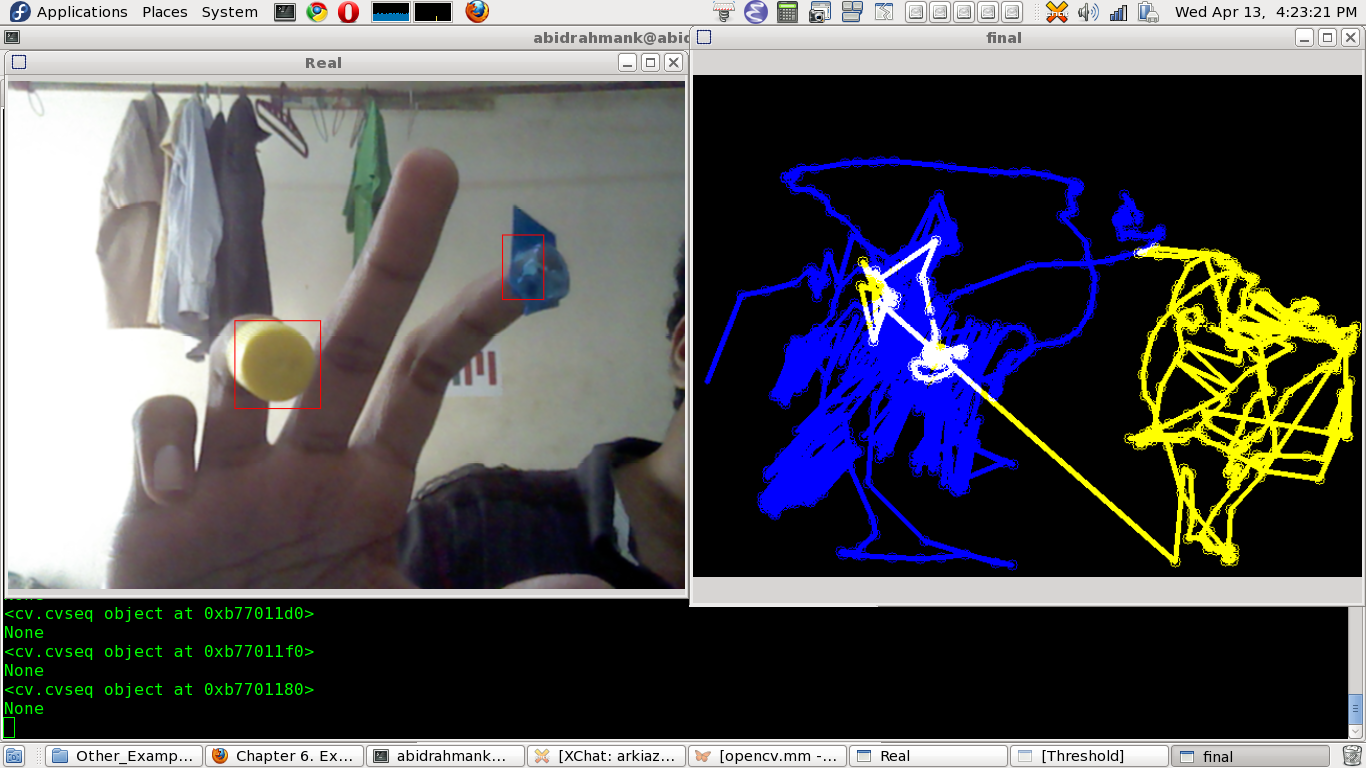
The second major release of the OpenCV was in October 2009. OpenCV 2 includes major changes to the [C++](https://en.wikipedia.org/wiki/C%2B%2B) interface, aiming at easier, more type-safe patterns, new functions, and better implementations for existing ones in terms of performance (especially on multi-core systems). Official releases now occur every six months and development is now done by an independent Russian team supported by commercial corporations.

In August 2012, support for OpenCV was taken over by a non-profit foundation OpenCV.org, which maintains a developer and user site.

On May 2016, Intel signed an agreement to acquire Itseez, a leading developer of OpenCV.

In July 2020, OpenCV announced and began a Kickstarter campaign for the [OpenCV AI Kit](https://opencv.org/introducing-oak-spatial-ai-powered-by-opencv/), a series of hardware modules and additions to OpenCV supporting Spatial AI.

Object Detection- Object detection is a computer vision technique in which a software system can detect, locate, and trace the object from a given image or video. The special attribute about object detection is that it identifies the class of object (person, table, chair, etc.) and their location-specific coordinates in the given image. The location is pointed out by drawing a bounding box around the object. The bounding box may or may not accurately locate the position of the object. The ability to locate the object inside an image defines the performance of the algorithm used for detection. Face detection is one of the examples of object detection.



These object detection algorithms might be pre-trained or can be trained from scratch. In most use cases, we use pre-trained weights from pre-trained models and then fine-tune them as per our requirements and different use cases.

**Chapter 2**

**Literature Review**

Digital art using machine learning has revolutionized the entire stream of learning and has notably spread to the lengths and breadth of various domains of learning. Machine learning is used as a tool of deep learning which is able to operate on vast amount of row, high dimensional data to learn hierarchies of representations. Renowned artists are making use of machine learning to give new dimensions to their art and upgrade the artist within them. Artist like Robert Thomas uses algorithmic processing of smartphone sensors to transform musical composition.

In, ‘Handwritten Text Recognition using Machine Learning Techniques in Application of NLP: This paper proposes Handwriting Detection is a technique or ability of a computer to receive and interpret intelligible handwritten input from source such as paper documents, touch screen, photo graphs etc. Handwritten Text recognition is one of area pattern recognition. The purpose of pattern recognition is to categorizing or classification data or object of one of the classes or categories. Handwriting recognition is defined as the task of transforming a language represented in its spatial form of graphical marks into its symbolic representation. Each script has a set of icons, which are known as characters or letters, which have certain basic shapes. The goal of handwriting is to identify input characters or image correctly then analysed to many automated process systems. This system will be applied to detect the writings of different format. The development of handwriting is more sophisticated, which is found various kinds of handwritten character such as digit, numeral, cursive script, symbols, and scripts including English and other languages. The automatic recognition of handwritten text can be extremely useful in many applications where it is necessary to process large volumes of handwritten data, such as recognition of addresses and postcodes on envelopes, interpretation of amounts on bank checks, document analysis, and verification of signatures. Therefore, computer is needed to be able to read document or data for ease of document processing. In, ‘Handwritten Text Recognition: With Deep Learning and Android: This research paper offers a new solution to traditional handwriting recognition techniques using concepts of Deep learning and computer vision. An extension of MNIST digits dataset called the Emnist dataset has been used. It contains 62 classes with 0-9 digits and A-Z characters in both uppercase and lowercase. An application for Android, to detect handwritten text and convert it into digital form using Convolutional Neural Networks, abbreviated as CNN, for text classification and detection, has been created. Prior to that we pre-processed the dataset and applied various filters over it. We designed an android application using Android Studio and linked our handwriting text recognition program using tensorflow libraries. The layout of the application has been kept simple for demonstration purpose. It uses a protobuf file and tensorflow interface to use the trained keras graph to predict alphanumeric characters drawn using a finger.

In, Inertial Pen Based Alphabet Recognition using KNN Classifier: In today’s electronics world human machine interface is important part. Pen with inbuilt inertial sensors devices capture human handwriting or drawing motions in real-time and use the sensor data for recognition. An inertial sensor based Inertial pen consist of an inertial sensor MPU 9150(accelerometer gyroscope and magnetometer), microcontroller, and a wireless transmission module, for sensing and collecting movement data for writing alphabet. The sensor data is received and processed for alphabets, recognition. The recognition algorithm composes of the steps of sensor data acquisition, signal pre-processing, feature generation, feature selection, and classification. KNN Classifiers for classification among 26 capital alphabets classes is built. The project aims at to validate the effectiveness of the inertial pen-based motion data acquisition and recognition of class of test sample from among 26 classes. The recognition accuracy achieved is 82%. The recognition accuracy of 93% is achieved for recognition of four gestures.

In, ‘An Inertial Pen with Dynamic Time Warping Recognizer for Handwriting and Gesture Recognition: This paper presents an inertial-sensor based digital pen (inertial pen) and its associated dynamic time warping (DTW)-based recognition algorithm for handwriting and gesture recognition. Users hold the inertial pen to write numerals or English lowercase letters and make hand gestures with their preferred handheld style and speed. The inertial signals generated by hand motions are wirelessly transmitted to a computer for online recognition. The proposed DTW-based recognition algorithm includes the procedures of inertial signal acquisition, signal pre-processing, motion detection, template selection, and recognition. We integrate signals collected from an accelerometer, a gyroscope, and a magnetometer into a quaternion based complementary filter for reducing the integral errors caused by the signal drift or intrinsic noise of the gyroscope, which might reduce the accuracy of the orientation estimation. Furthermore, we have developed a minimal intraclass to maximal inter-class-based template selection method (min-max template selection method) for a DTW recognizer to obtain a superior class separation for improved recognition. Experimental results have successfully validated the effectiveness of the DTW-based recognition algorithm for online handwriting and gesture recognition using the inertial pen.

There are different drawbacks of earlier implemented systems like some are very less accurate. Other one system required very bulky hardware like sensors and wearable gloves for hand gesture recognition which becomes very uncomfortable after some time. Some systems have very complex design. Hence to overcome this we developed the model of air writing based on machine learning language in which we used python and OpenCV.

Machine Learning applications are becoming an integral part of our life and for the surrounding society. Real life applications in the field of machine learning can improve the efficiency and consistency in our day-to-day work, by using computer vision and its associating algorithms and its subsequent modules or libraries can help us develop this type of real-life applications. OpenCV library is an open-source library for building a machine learning architecture that can track and process real life data and manipulate it for a required set of expected output. Related research on various papers and its summaries are as follows: Title of the Paper is Hand Gesture Recognition its authors are Rafiqul Zaman Khan, Noor Adnan Ibraheem and the related paper’s essential aim of building hand gesture recognition system is to create a natural interaction between human and computer where the recognized gestures can be used for controlling a robot or conveying meaningful information. How to form the resulted hand gestures to be understood and well interpreted by the computer considered as the problem of gesture interaction Human computer interaction (HCI) also named Man-Machine Interaction. Title of the Paper is Gesture Controlled Computer its authors are S. T. Gandhe, Nikita A. Pawar, Mayuri S. Hingmire, Kalpesh P. Mahajan, Devshri V. Patil. This paper describes the method for humans to interact with digital world and use the computer with just our hand movements. The paper is based on image processing. The camera detects gestures and converts those gestures into equivalent digital algorithms as programmed. This paper deals with the controlling all operations of mouse such as right click, left click and movement of cursor over the desktop, drag and drop, snapshot, Air writing and painting through hand gestures.

**Chapter3**

**Methodology**

**3.1**

**3.1.1 Approaches in Object detection:**

In this section, we will briefly go through the different approaches that are taken in Object detection tasks. There are two approaches to Object detection and they are:

1. Two-shot detection.
2. Single-shot detection.

Let us first find about Two-shot detection method. As the name suggests there are two stages involved in this method. One is region proposal and then in the second stage, the classification of those regions and refinement of the location prediction takes place.

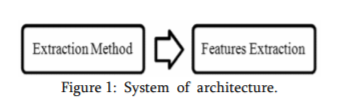
Faster-RCNN variants are the popular choice of usage for two-shot models. Here during the region proposal stage, we use a network such as ResNet50 as a feature extractor. We do this by removing the last layers of this network and just use the rest of the layers to extract features from the images. This is usually a better approach as the network is already trained and can extract features from the images. Next, a small fully connected network slides over the feature layer to predict class-agnostic box proposals, with respect to a grid of anchors tiled in space, scale and aspect ratio.

In the second stage, these box proposals are used to crop features from the intermediate feature map which was already computed in the first stage. The proposed boxes are fed to the remainder of the feature extractor in which the prediction and regression heads are added on top of the network. Finally, in the output, we get the class and class-specific box refinement for each proposal box.

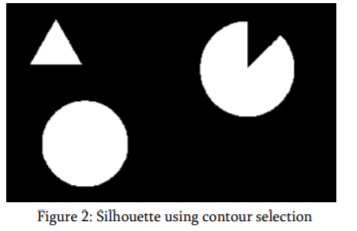
On the contrary side, Single-shot detection skips the region proposal stage and yields final localization and content prediction at once. YOLO is a popular example of this approach and we are going to discuss the working of it in the coming sections.

It must be noted that two-shot detection models achieve better performance but single-shot detection is in the sweet spot of performance and speed/resources which makes it more suitable for tasks like detecting objects in live feed or object tracking where the speed of prediction is of more importance.

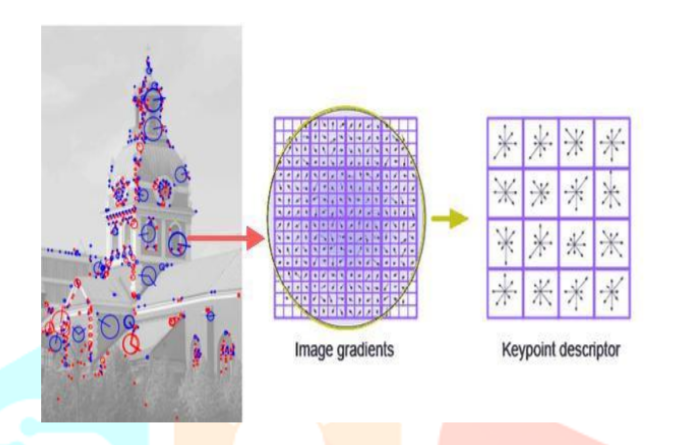
**3.1.2 The proposed system**

Proposed system can be classified into mainly two steps after acquiring the input image from camera, videos or even an Object of Interest. These steps are: Extraction Method image pre-processing and Feature’s estimation and Extraction. 

1. *Extraction Method and image pre-processing:* Segmentation process is the first process for recognizing Object of Interest. It is the process of dividing the input image (in this case Object of Interest image) into regions separated by boundaries. The segmentation process depends on the type of object, if it is dynamic object then the Object need to be located and tracked, if it is static object with no moving parts such as a coloured bottle cap or a pen of particular colour the input image has to be segmented only. The object should be located firstly, generally a bounding box is used to specify the depending on the object colour and secondly, the object have to be tracked, for tracking the object there are two main approaches either the video is divided into frames and each frame have to be processed alone, in this case the object frame is treated as a moving OBJ(Object-of-Interest) and segmented, or using some tracking information such as shape, object colour using some filter. The common helpful cue used for segmenting the object is its colour, since it is easy and invariant to scale, translation, and rotation changes. In this case a coloured markers which provide exact information about the orientation and position. The color space used in a specific application plays an essential role in the success of segmentation process, however colour spaces are sensitive to lighting changes.
2. *Features Extraction*- Good segmentation process leads to perfect features extraction process and the latter play an important role in a successful recognition process. The modules of OpenCV for image processing applications are given below: CORE module contains basic data structures and basic functions used by other modules. IMGPROC module contains image processing related functions such as linear, non-linear image filtering and geometrical image transformations etc. VIDEO module contains motion estimation and object tracking algorithms. ML module contains machine-learning interfaces. HighGUI module contains the basic I/O interfaces and multi-platform windowing capabilities. Features vector of the segmented image can be extracted in different ways according to particular application. Various methods have been applied for representing the features can be extracted. In this method we have used the shape of the object such as object contour. A contour is a closed curve of points or line segments that represents the boundaries of an object in the image. Contours are essentially the shapes of objects in an image. Contours are sometimes called a collection of points or line segments that overall represent the shape of the object in an image.



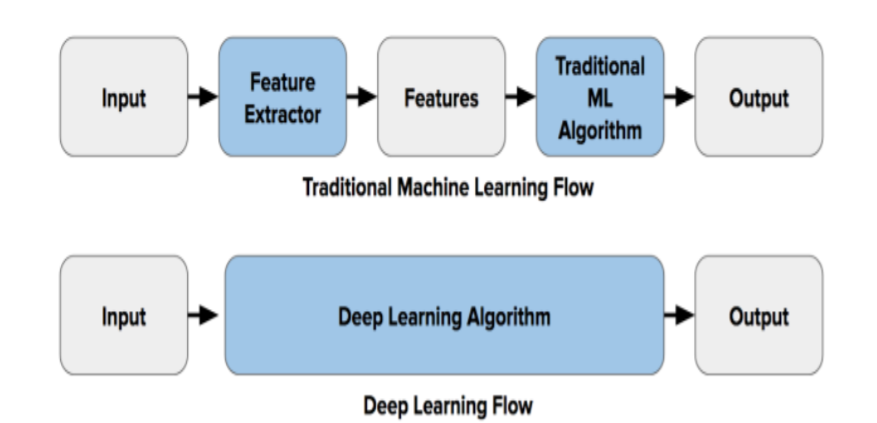
Computer vision can be succinctly described as finding and telling features from images to help discriminate objects and/or classes of objects. Computer vision has become one of the vital research areas and the commercial applications bounded with the use of computer vision methodologies is becoming a huge portion in industry. The accuracy and the speed of processing and identifying images captured from cameras are has developed through decades. Being the well-known boy in town, deep learning is playing a major role as a computer vision tool. Computer vision was mainly based with image processing algorithms and methods. The main process of computer vision was extracting the features of the image. Detecting the color, edges, corners and objects were the first step to do when performing a computer vision task. These features are human engineered and accuracy and the reliability of the models directly depend on the extracted features and on the methods used for feature extraction. In the traditional vision scope, the algorithms like SIFT (Scale-Invariant Feature Transform), SURF (Speeded-Up Robust Features), BRIEF (Binary Robust Independent Elementary Features) plays the major role of extracting the features from the raw image. The difficulty with this approach of feature extraction in image classification is that you have to choose which features to look for in each given image. When the number of classes of the classification goes high or the image clarity goes down it’s really hard to cope up with traditional computer vision algorithms.



**3.2**

**3.2.1 Proposed Algorithm:**

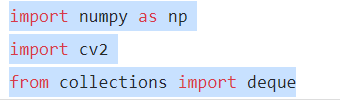
Deep learning, which is a subset of machine learning has shown a significant performance and accuracy gain in the field of computer vision. Arguably one of the most influential papers in applying deep learning to computer vision, in 2012, a neural network containing over 60 million parameters significantly beat previous state-of-the-art approaches to image recognition in a popular Image Net computer vision competition. The boom started with the convolutional neural networks and the modified architectures of ConvNets. By now it is said that some convNet architectures are so close to 100% accuracy of image classification challenges, sometimes beating the human eye.

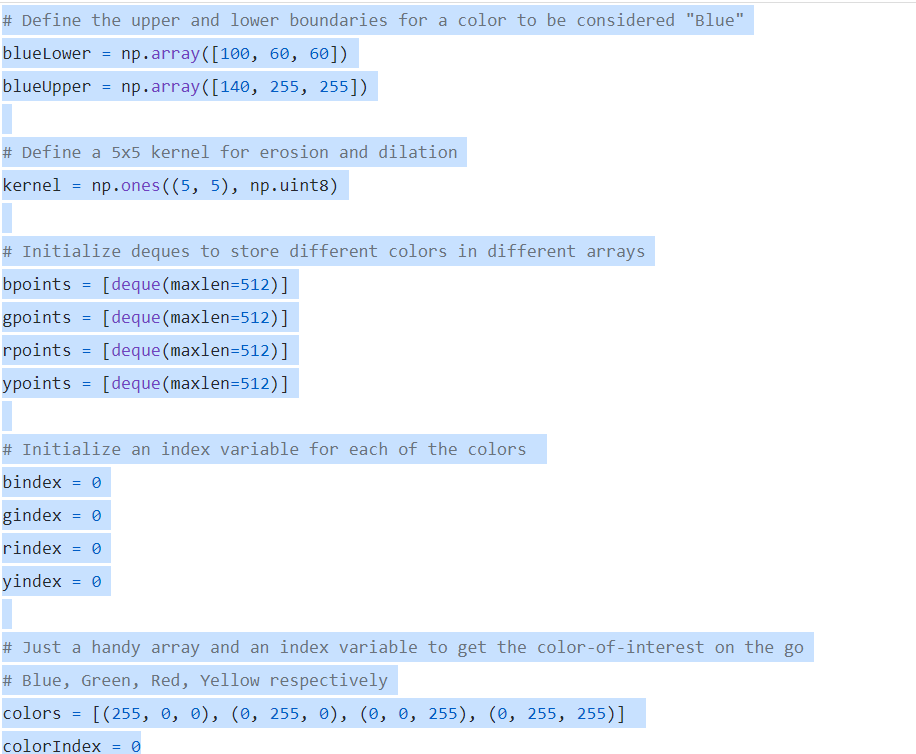
****

# 3.2.2 Code Explanation

## Step 1: Initialize Some Stuff

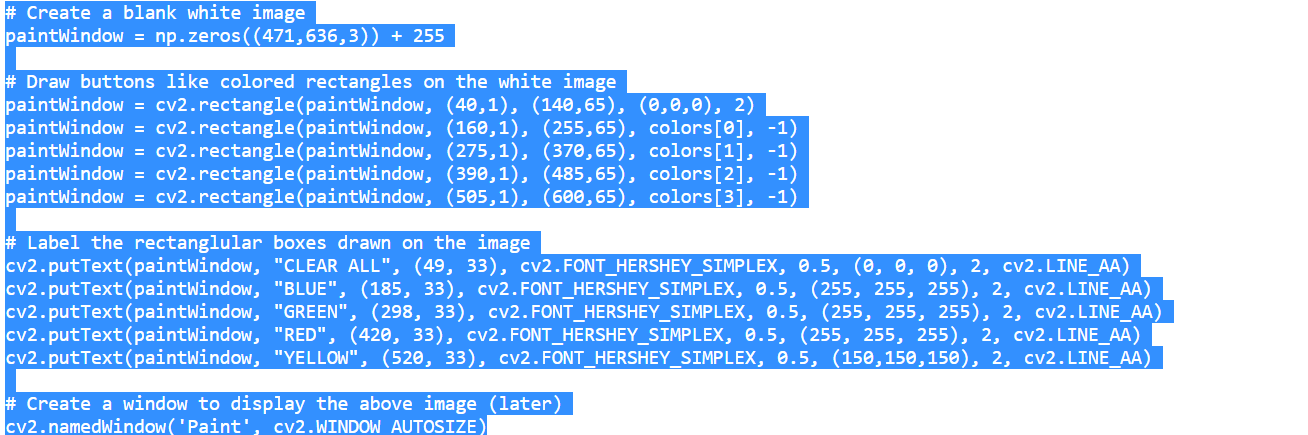
Firstly, we import the necessary libraries.



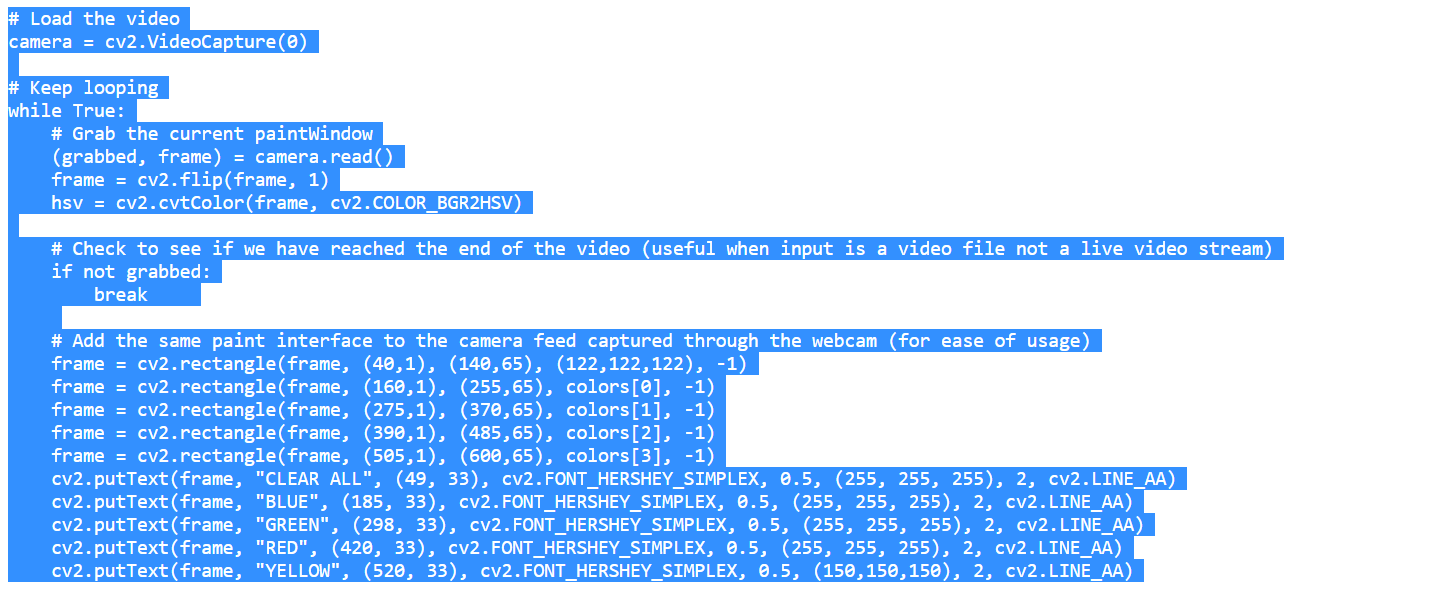
Then we initialize variables that are used in the following steps. The blueLower and the blueUpper numpy arrays help us in finding the blue colored cap. The kernal helps in smoothing blue cap once found. The bpoints, gpoints, rpoints and ypoints deques are used to store the points drawn on the screen of color blue, green, red and yellow respectively.

## Step 2: Setup the Paint Interface

This is a painful thing to do. We have to manually set the coordinates of each of the color boxes on the frame. We use the OpenCV function **cv2.rectangle()** to draw the boxes.

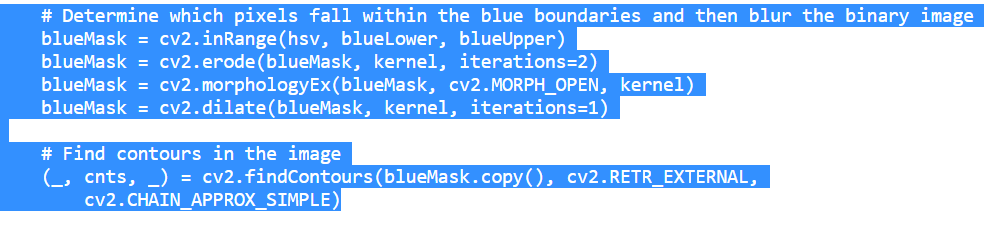


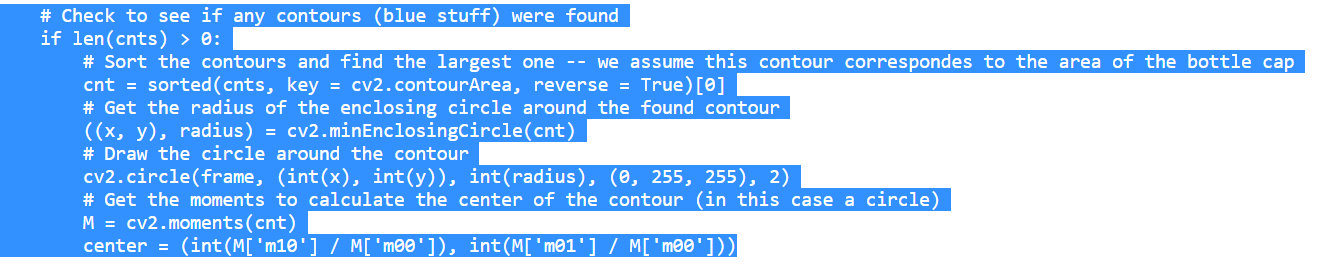
## Step 3: Start Reading the Video (Frame by Frame)

Now we use the OpenCV function **cv2.VideoCapture()**method to read a video, frame by frame (using a while loop), either from a video file or from a webcam in real-time. In this case, we pass 0 to the method to read from a web.We can just add the exact same paint interface for ease of usage.

## Step 4: Find the Contour-Of-Interest (The Pen/Bottle Cap)

Once we start reading the webcam feed, we constantly look for a blue color object in the frames with the help of **cv2.inRange()** method and use the blueUpper and blueLower variables initialized in Step 0. Once we find the contour, we do a series of image operations and make it smooth. They just makes our process easier!

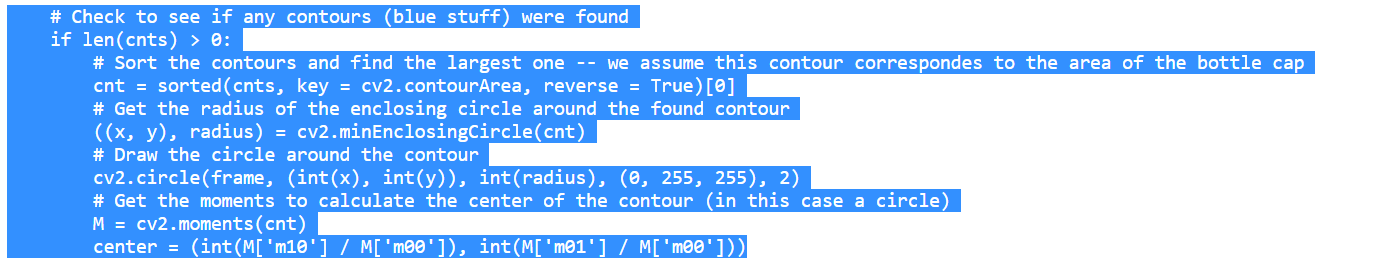
Once we find the contour (the **if** condition passes when a contour is found), we use the center of the contour (blue cap) to draw on the screen as it moves. The following code does the same.



The above code finds the contour (the largest one), draws a circle around it using the **cv2.minEnclosingCircle()** and **cv2.circle()** methods, gets the center of the contour found with the help of **cv2.moments()** method.

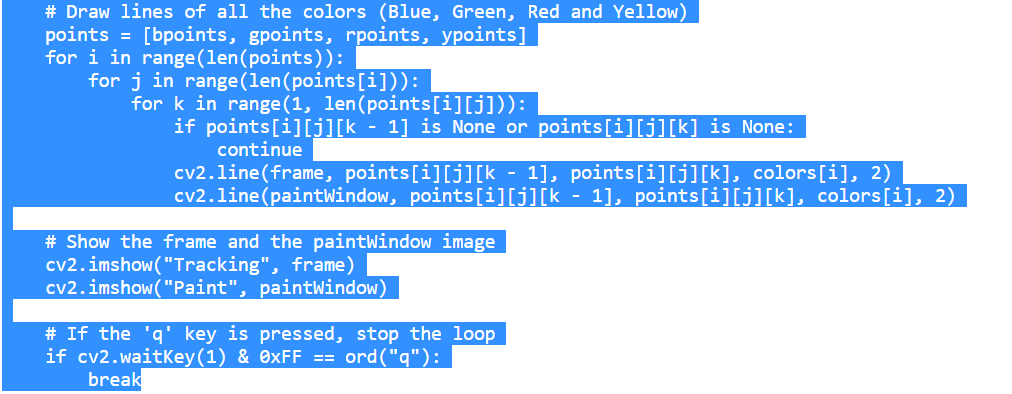
## Step 5: Start Drawing and Store the Drawings

Now we start tracking coordinates of each and every point the center of the contour touches on the screen, along with its color. We store these set of points of different colors in different deques (bpoints, gpoints etc.). When the center of the contour touches one of the colored boxes we put on the screen in Step 1, we store the points in its respective color deque.



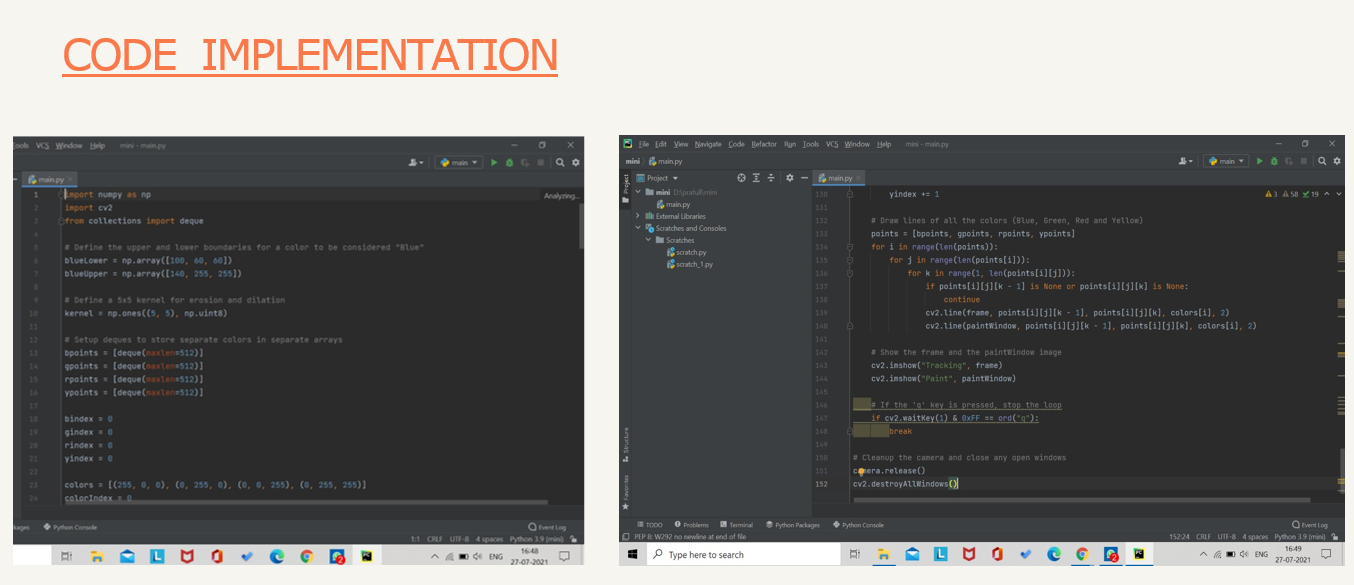
**Step 6: Show The Drawings On The Screen**

So far we stored all the points in their respective color deques. Now we just join them using a line of their own color. The OpenCV function **cv2.line()** comes in handy for us to do that. The following code does the same.



Once we join all the points in each and every frame with a line and put it on both the windows, we created using **cv2.imshow()** method and it all fits perfectly to work like a paint application. After falling out of the **while** loop, we entered to read data from the webcam, we release the camera and destroy all the windows using the following lines of code.

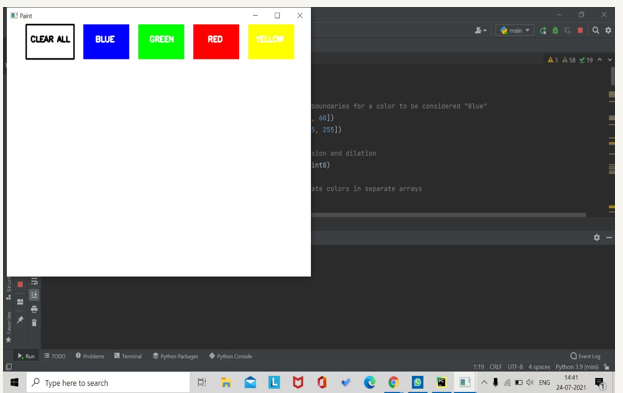


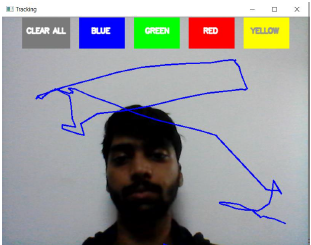


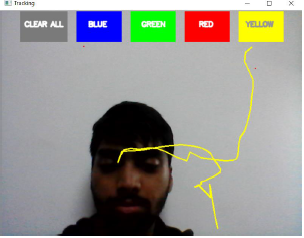
That’s it! We successfully used a bunch of basic image processing tools and operations in OpenCV to create a paint-like application —Webcam Paint and now we can draw in air while in front of webcam!

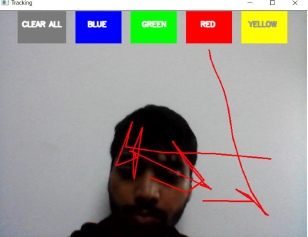
**Chapter 4**

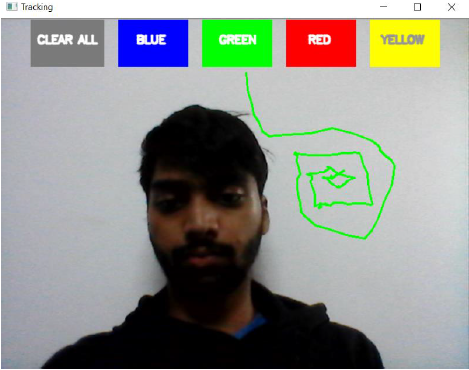
**Experimental Results**

****

****

****

****

****

**Chapter 5**

**Conclusion**

**5.1**

A branch of artificial intelligence and image processing concerned with computer processing of images from the real world. Computer vision can be described as finding and telling features from images to help discriminate objects and/or classes of objects. The difficulty with this approach when the number of classes of the classification goes high or the image clarity goes down it’s really hard to cope up with traditional computer vision algorithms. Hence, we can conclude that Deep learning, which is a subset of machine learning has shown a significant performance and accuracy gain in the field of computer vision.

Demonstration of the image processing capabilities of OpenCV. The ultimate goal is to create a computer vision machine learning application that promotes Human computer interaction (HCI) also named Man Machine Interaction (MMI)] refers to the relation between the human and the computer or more precisely the machine, and since the machine is insignificant without suitable utilize by the human there are two main characteristics should be deemed when designing a HCI system as mentioned: functionality and usability. System functionality referred to the set of functions or services that the system equips to the users while system usability referred to the level and scope that the system can operate and perform specific user purposes efficiently.

**5. 2 Real World Application of our project:**

Our project demonstrates how we can draw in air while in front of camera, which has a variety of uses, some of which are-

* Online Teaching as any surface can be used as writing board.
* Visualization will better be using a 3D whiteboard
* Human-computer interaction
* Security and surveillance
* Video communication and compression
* Augmented reality, traffic control, medical imaging and video editing etc.

**vi. References**

1. [**https://www.ijcrt.org/papers/IJCRT2006458.pdf**](https://www.ijcrt.org/papers/IJCRT2006458.pdf)
2. [**https://ijsrst.com/paper/7305.pdf**](https://ijsrst.com/paper/7305.pdf)
3. [**https://towardsdatascience.com/tutorial-webcam-paint-opencv-dbe356ab5d6c**](https://towardsdatascience.com/tutorial-webcam-paint-opencv-dbe356ab5d6c)
4. [**https://jespublication.com/upload/2020-1107165.pdf**](https://jespublication.com/upload/2020-1107165.pdf)
5. [**https://www.geeksforgeeks.org/live-webcam-drawing-using-opencv/**](https://www.geeksforgeeks.org/live-webcam-drawing-using-opencv/)
6. [**https://en.wikipedia.org/wiki/Computer\_vision**](https://en.wikipedia.org/wiki/Computer_vision)
7. Real Time Object Detection and Tracking Using Deep Learning and OpenCV Proceedings of the International Conference on Inventive Research in Computing Applications (ICIRCA 2018) IEEE Xplore Compliant Part Number: CFP18N67-ART; ISBN:978-1-5386-2456-2
8. International journal of Computer Vision by Yasuyuki Matsushita, Jiri Matas, Svetlana Lazebnik