A Pointing Gesture-based Signal to Text Communication System Using OpenCV in Python

Saachi Shrikhande

Department of Electronics and Telecommunications, Vishwakarma Institute of Technology, Pune

---***---

Abstract — Air canvas helps to draw on a screen just by waving your finger fitted with a colourful point or a simple coloured cap. The paper uses the computer vision techniques of OpenCV to study the system. The paper is written based on a system in python due to its exhaustive libraries and simple syntax but after understanding the basics it can be implemented in any OpenCV-supported language.

Index terms: Open Computer Vision, object tracking, masking, hand gesture, optical character reorganization

I. INTRODUCTION

This paper presents a Object tracking is a field of Computer Vision which has been brought into being by the availability of faster computers, cheaper and better cameras and demands for automated video analysis.[1] The video analysis procedure has three main steps: detecting the object, tracking its movement from frame to frame and analysis of the behaviour of that object. There can be 4 things that can cause a problem when object tracking is taken into consideration: selection of a suitable object, feature selection, object detection and object tracking. Object tracking algorithms are one of the most important parts of applications like automatic surveillance, video indexing vehicle navigation etc. Human-computer interaction can be performed with object tracking. With the expanding boom of augmented reality, such interaction seems of the top priority. The alternatives for human-computer interactions come down to two ways: images based and glove based. The first approach needs a dataset of images to recognize hand movements while with glove based approach we special sensors as hardware. The glove-based applications can be very useful for specially-abled individuals.[5]

In this paper, instantaneous, real-time pointing gesture tracking and recognition using a video input algorithm are discussed. The focus was on first, detecting the coloured tip of a finger in the video feed and then applying optical character reorganization(OCR) to recognize the intended text. The rest of the paper is divided into six sections. Section 2-4 discusses the other uses of the concept of human-computer interaction, and the suggested method and discusses the results. Section 5 concludes our findings and suggests betterment in the current approach.[7]

II. LITERATURE REVIEW

When looking deeper into finger tracing one can observe that this can be approached in many different ways. One group of researchers used kinetic sensors to detect the hand shape in some systems. Although hand gestures were still a very challenging problem, the kinetic sensor helped with depth modelling, and noise elimination among many other advantages. The resolution of this Kinect sensor in one of the systems studied was 640×480 which works well to track a large object like a human body but tracking merely a finger is difficult.[3, 21]

Another set of researchers implemented finer tracing into hardware which they called smart glasses. These helped the user to experience egocentric interaction with a set of simple, modified glasses. The hand gesture is fed into the data system and measures are taken to allow the system to work in the background with multiple colours which might disrupt object tracking.[1, 5]

III. METHODOLOGY

In this section, the suggested method to go about gesture-to-text conversion model.

When the user traces a pattern the finger with which the patternis created, is traced and a trajectory is created. The layers to the process is explained in Figure 1. Basically when a finger is detected the system starts reading the frames and converts the captured frames to HSV color space. This conversion is essential because this helps us foe better colour detection.

Once this is done the canvas frame is prepared and the respective ink buttons are put on it. This is followed by adjusting the track bar values for finding the mask of the coloured marker. Preprocessing of the mask is done next, with morphological operations; namely erosion and dilation. Detecting the contours, finding the centre coordinates of the largest contour and storing them in the array for successive frames follows preprocessing. Finally, the points stored in an array are drawn on the frames and canvas.

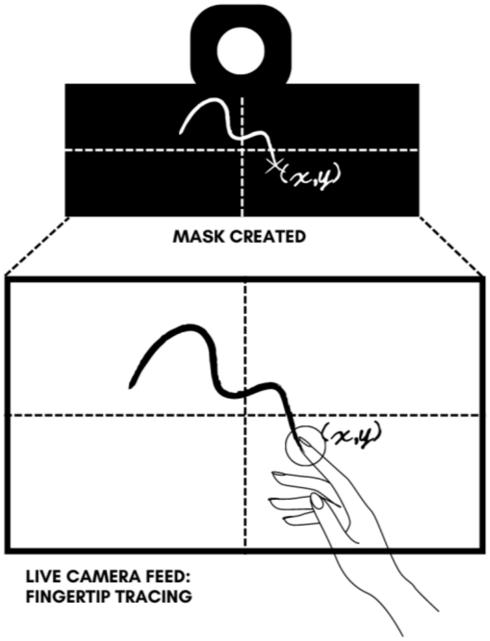

Figure 1: Workflow of the system

If we want to dive deeper into the working of such a system, we have to understand the concept of OpenCV.

OpenCV is a large open-source of computer vision, machine learning, and image processing. OpenCV supports various editing languages such as Python, C ++, Java, etc.

It can process photos and videos to see objects, faces, or even handwriting.

When combined with a variety of libraries, such as Numpy which is a well-designed library of numerical uses, the number of weapons grows in your Arsenal that is, any work one can do at Numpy can be integrated with OpenCV.

If we were to dive deeper into computer vision, almost all computer vision algorithms use neural networks, a powerful machine learning algorithm. A neural network is composed of neurons, also known as units. Figure 2 shows the structure of a neural network.

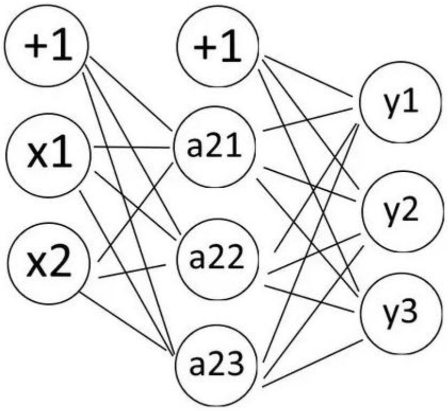

Figure 2: General structure of a neural network

The idea is as follows: You have input neurons which make up the input layer. Each input neuron performs an operation on the value it received, and sends it each neuron in the next layer. Each of the neurons in the next layer connects to each neuron in the layer after that, and so on until you get to the output layer. Once you hit the output layer, you have successfully evaluated the neural network. This algorithm forwards propagation. One thing to note is that each layer has an added bias neuron, which always outputs +1.

The sigmoid(because if you graph it, it looks like an S) function is used in neural networks. The equation for the same is given as Equation 1.

$$g(z) = 1/(1 + e^{-z}) \tag{1}$$

where

g(z) = sigmoid function

e = euler's constant

Graphically we can represent this as shown in Figure 3.

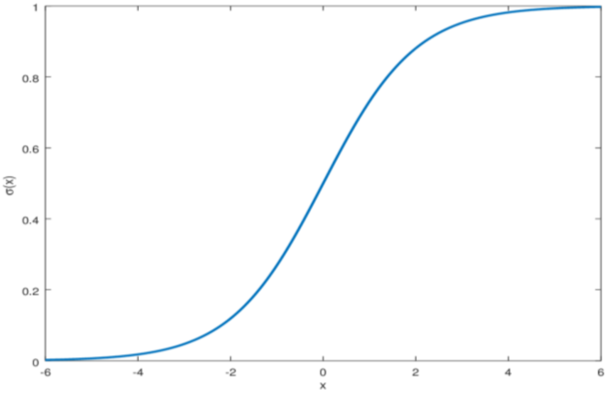

Figure 3: The elongated 'S'-like curve of the sigmoid function

Each connection between neurons has a value associated with it, known as a weight. If you think about it, you may realize that we can actually store the weights for each layer l in a separate matrix, Theta_l.

Now that we're using matrices, let's also have our X as a column vector. These can be multiplied.

$$\boldsymbol{al} + \mathbf{1} = \boldsymbol{g}(\Theta \boldsymbol{al} * \boldsymbol{al}) \qquad (2)$$

For each layer l, evaluate the Equation 2 until the last layer is reached.

One important thing is that a_l must be a column vector with elements equal to the number of neurons in that layer.

That means, that if we have K classes among which we can classify something, we will have a K-dimensional column vector as our output vector. This vector contains the probability of each class. We usually store this output vector as h(x), or the hypothesis. The next thing that we should be aware of is Erosion and Dilation of images. Morphological works are a set of tasks that process images based on conditions. They apply a structural element to an input image and produce an output image.

The most basic morphological functions are two: Erosion and Ascension. Basic erosion is used to reduce image features. It removes the restrictions of the previous item. Erosion Performance checks some features which as can be defined as: The kernel (matrix of odd size (3,5,7) is integrated with the image. The pixel in the first image (either 1 or 0) will be considered as 1 only if all pixels below 1 kernel, otherwise, are eroded (made to zero). So all pixels near the border will be discarded depending on the size of the kernel and the thickness or size of the front element decreases or the white region decreases in the image.

The basis for expansion is that it increases the object area and is used to emphasize the features.

The kernel (matrix of odd size (3,5,7) is integrated with the image.

The pixel fraction in the first image is '1' if at least one pixel below the kernel is '1'.

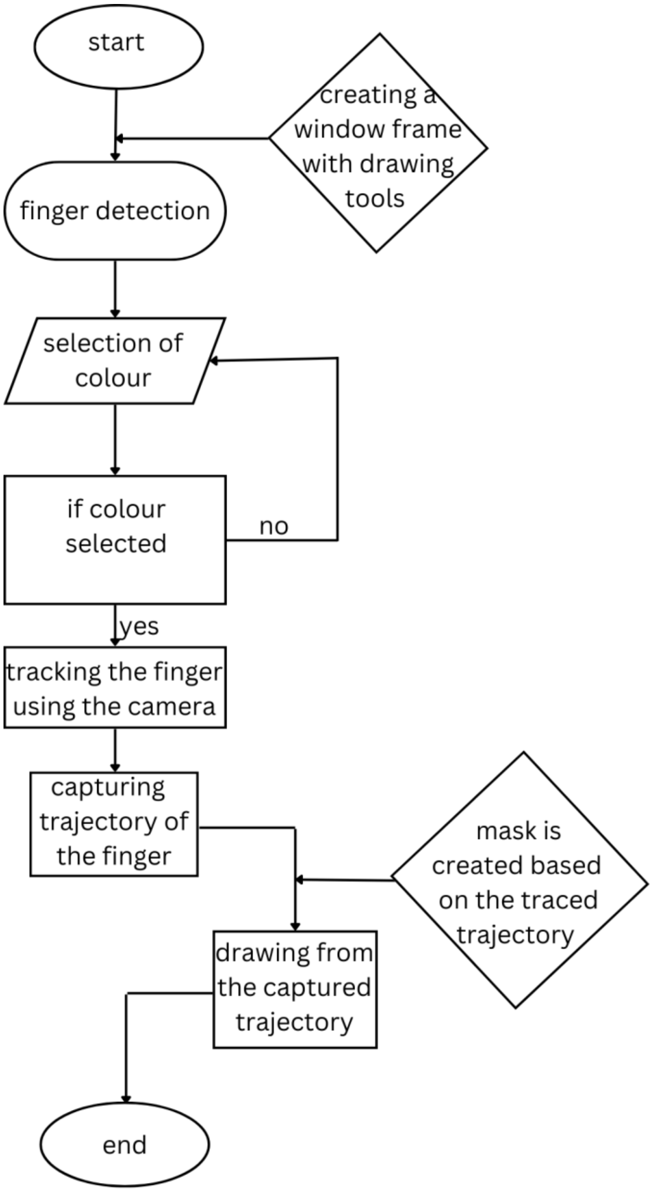
start
creating a window frame with drawing tools
finger detection
selection of colour
if colour selected
no
yes
tracking the finger using the camera
capturing trajectory of the finger
mask is created based on the traced trajectory
drawing from the captured trajectory
end

Figure 4: Flowchart of working of air canvas

Dilation effect increases the white area in the image or the size of the front object increases. The colour marker is tracked and a mask window is made. Mask is responsible for the processes, of erosion and dilution.

Erosion reduces disturbances in the mask whereas dilation is responsible for restoring any main tracked part removed.

Figure 4 shows how all this is executed sequentially.

IV. RESULTS AND DISCUSSIONS

Virtual reality applications are our primary means of interacting with the environment. Although inverse kinematics (IK) tag-based motion sensing works for body tracking, it is less reliable for fingers, which are often occluded by cameras.

Many computer vision and virtual reality applications circumvent this problem by using an additional system (e.g., inertial trackers). Here we investigated an alternative solution that tracks hands and fingers only using a motion sensing system based on cameras and active tags with machine learning techniques. Our finger animation is performed by a predictive model based on neural networks, which is trained on a set of motion data obtained from several subjects with a complementary sensing system (inertial). The system is as effective as the traditional IK algorithm, providing natural pose reconstruction and resolving occlusions

The system that we have studied will give us a basic output of the application of OpenCV. The existing system only works with your fingers, with no highlighters, paints, or relatives. Identifying and interpreting a finger-like object from an RGB image without a deep sense sensor is a major challenge. The system uses a single RGB camera to record up. Since sensor depth is not possible, the movement of the pen up and down cannot be tracked. Therefore, the whole fingerprint trace is traceable, and the resulting image can be realistic and can be seen by the model.

V. CONCLUSION

Air Canvas is a program that helps you write anywhere just with the help of a coloured marker which is tracked by a camera. It aims to challenge traditional writing methods. Rather than carrying a notepad or your mobiles, air canvas provides a very convenient alternative. It will also help the specially-abled part of our society access technology easily. Not just them but even people who find difficulty dealing with technology will be able to use air canvas effortlessly. Drawing in the air can be converted from a science fiction concept to a real-life application. The proposed method includes two main tasks: tracking the coloured fingertip in the video frames and using English OCR on rendered images for recognition of written characters. Additionally, the proposed method provides a natural human-system interaction that does not require a keyboard, pen, glove, or other character device input. It only requires a mobile camera and the color red reorganization of the fingertip. However, it has one serious problem: it is colour sensitive in that there is any coloured object in the background before starting and during the analysis can lead to false results.

REFERENCES

[1] Y. Huang, X. Liu, X. Zhang, and L. Jin, "A Pointing Gesture Based Egocentric Interaction System: Dataset, Approach, and Application," 2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Las Vegas, NV, pp. 370-377, 2016.

[2] P. Ramasamy, G. Prabhu, and R. Srinivasan, "An economical air writing system is converting finger movements to text using a web camera," 2016 International Conference on Recent Trends in Information Technology (ICRTIT), Chennai, pp. 1-6, 2016.

[3] Saira Beg, M. Fahad Khan and Faisal Baig, "Text Writing in Air," Journal of Information Display Volume 14, Issue 4, 2013

[4] Alper Yilmaz, Omar Javed, Mubarak Shah, "Object Tracking: A Survey", ACM Computer Survey. Vol. 38, Issue. 4, Article 13, Pp. 1-45, 2006

[5] Nishtha Dua, Akash Kumar Choudhary, Bharat Phogat, “Air Canvas using Numpy and OpenCV in Python” International Journal of Research in Engineering and Technology · August 2021

[6] Bach, Benjamin, et al. "Drawing into the AR-CANVAS: Designing embedded visualizations for augmented reality." Workshop on Immersive Analytics, IEEE Vis. 2017

[7] S. Bambach, S. Lee, D. J. Crandall, and C. Yu. “Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions.” In Proceedings of the IEEE International Conference on Computer Vision, pages 1949–1957, 2015.

[8] A.Betancourt, P. Morerio, L. Marcenaro, M.Rauterberg, and C. Regazzoni. “Filtering SVM frame-by-frame binary classification in a detection framework.” In Image Processing (ICIP), 2015 IEEE International Conference on, pages 2552–2556. IEEE, 2015.

[9] M.Bindemann. “Scene and screen centre bias early eye movements in scene viewing.” Vision Research, 2010.

[10] M. Cheng, N. J. Mitra, X. Huang, P. H. Torr, and S. Hu. “Global contrast based salient region detection. Pattern

Analysis and Machine Intelligence", IEEE Transactions on, 2015.

[11] S. Goferman, L. Zelnik-Manor, and A. Tal. "Context-aware saliency detection. Pattern Analysis and Machine Intelligence", IEEE Transactions on, 2012.

[12] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. "Highspeed tracking with kernelised correlation filters. Pattern Analysis and Machine Intelligence", IEEE Transactions on, 2015.

[13] Y. Huang, X. Liu, X. Zhang, and L. Jin. Deepfinger: "A cascade convolutional neuron network approach to finger key point detection in egocentric vision with a mobile camera." In The IEEE Conference on System, Man and Cybernetics (SMC), pages 2944–2949. IEEE, 2015.

[14] Z. Kalal, K. Mikolajczyk, and J. Matas. "Tracking-learning detection. Pattern Analysis and Machine Intelligence", IEEE Transactions on, 2012.

[15] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk. "Slic superpixels compared to state-of-the-art superpixel methods. Pattern Analysis and Machine Intelligence", IEEE Transactions on, 2012.

[16] A. Fathi and J. M. Rehg. "Modelling actions through state changes." In Proc. of CVPR, 2013.

[17] A. Fathi, X. Ren, and J. M. Rehg. "Learning to recognize objects in egocentric activities" In Proc. of CVPR, 2011.

[18] S. Vikram, L. Li, and S. Russell, "Handwriting and gestures in the air, recognizing on the fly," in Proceedings of the CHI, vol. 13, 2013, pp. 1179–1184.

[19] Martin de La Gorcel, Nikos Paragios and David J. Fleet. "Model-Based Hand Tracking with Texture, Shading and Selfocclusions." In Proc. of IEEE CVPR, 2008.

[20] Zhichao, Ye & Zhang, Xin & Jin, Lianwen & Feng, Ziyong & Xu, Shaojie. (2013). "Finger-writing-in-the-air system using Kinect sensor". Electronic Proceedings of the 2013 IEEE International Conference on Multimedia and Expo Workshops, ICMEW 2013

[21] Oludare Isaac Abiodun, Aman Jantan, Abiodun Esther Omolara Kemi Victoria Dada, Nachaat ABD Elatif Mohamed, Humaira Arshad, "State-of-the-art in artificial neural network applications: A survey", Heliyon, Volume 4, Issue 11, 2018

[22] D. Pavllo, T. Porssut, B. Herbelin and R. Boulic, "Real-Time Marker-Based Finger Tracking with Neural Networks," 2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR), 2018

[23] E. Hayman and J.-O. Eklundh. "Statistical background subtraction for a mobile observer." In Proc. of ICCV, 2003.

[24] S. Albawi, T. A. Mohammed and S. Al-Zawi, "Understanding of a convolutional neural network," 2017 International Conference on Engineering and Technology (ICET), 2017