GESTURE BASED WRITING USING COMPUTER VISION

A MINI PROJECT REPORT

Submitted by

Sudharsan R (142221104136)

Vikram V (142221104159)

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

SRM VALLIAMMAI ENGINEERING COLLEGE

(AN AUTONOMOUS INSTITUTION)

ANNA UNIVERSITY : CHENNAI 600 025

MAY 2024

BONAFIDE CERTIFICATE

Certified this project report "**GESTURE BASED WRITING USING COMPUTER VISION**" is the bonafide work of **"SUDHARSAN.R (142221104136) and VIKRAM.V (142221104159)"** who carried out the work under my Supervision.

SIGNATURE	**SIGNATURE**
Dr. M. MAYURANATHAN, M.Tech., Ph.D.,	**Dr. B. VANATHI, M.E., Ph.D.,**
ASSOCIATE PROFESSOR,	**PROFESSOR & HEAD,**
Department of CSE,	Department of CSE,
SRM Valliammai Engineering College,	SRM Valliammai Engineering College,
Kattankulathur – 603 203.	Kattankulathur – 603 203.

Submitted for the university examination held on____________________at
SRM Valliammai Engineering College, Kattankulathur.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We sincerely express our gratitude in depth to our esteemed Founder Chairman & Chancellor **Dr. T. R. Paarivendhar, Thiru. Ravi Pachamoothoo** Chairman SRM Group & Pro-Chancellor (Admin). **Mrs. R. Harini,** Correspondent, SRM VEC and authorities of **SRM VALLIAMMAI ENGINEERING COLLEGE** for the patronage on our welfare rooted in the academic year.

We express our sincere gratitude to our respected **Director, Dr. B. Chidhambararajan, M.E., Ph.D.,** for his constant encouragement, which has been motivated to strive towards excellence. We thank our sincere **Principal Dr. M. Murugan, M.E., Ph.D.,** for his constant encouragement, which has been our motivation to strive towards excellence.

We extend our hand of thanks to our Head of the Department, **Dr. B. Vanathi, M.E., Ph.D.,** Professor for her instituted support.

We are grateful to our internal guide **Dr. M. Mayuranathan, M.Tech., Ph.D.,** Associate Professor, without his invaluable guidance and encouragement, this mini project would not have been a success.

We also like to thank all Teaching and non-teaching staff members of our department, for their support during the mini project.

We finally thank our friends and family for their support during the mini project.

ABSTRACT

The Gesture based writing method described is a real-time video-based pointing system that enables users to sketch and write English text in the air using a coloured fingertip. The colour detection & tracking technique used in this method allows the software to track the coloured fingertip & produce a mask, which is then subjected to morphological operations such as erosion & dilation.

Erosion is a morphological operation that shrinks the mask, effectively reducing impurities present in the image, which helps to clean up the image and remove any small or unwanted features that may interfere with the tracking process. On the other hand, dilation is the opposite of erosion & works to expand the mask, which helps to restore the eroded areas of the main mask.

The method can be useful for a variety of applications, such as virtual whiteboarding, remote teaching, or teleconferencing & also used by disabled person who wants to write while teaching, where users can easily write or sketch ideas without the need for a physical writing surface.

Furthermore, the technique is relatively low cost & accessible, requiring only a camera & a coloured marker, making it a potential alternative to more expensive & complex pointing devices.

ABSTRACT (TAMIL)

விவரிக்கப்பட்டுள்ள சைகை அடிப்படையிலான எழுத்து முறையானது நிகழ்நேர வீடியோ அடிப்படையிலான சுட்டி அமைப்பாகும், இது பயனர்கள் வண்ண விரல் நுனியைப் பயன்படுத்தி காற்றில் ஆங்கில உரையை வரைவதற்கும் எழுதுவதற்கும் உதவுகிறது. இந்த முறையில் பயன்படுத்தப்படும் வண்ணக் கண்டறிதல் மற்றும் கண்காணிப்பு நுட்பம் மென்பொருளை வண்ண விரல் நுனியைக் கண்காணிக்கவும் முகமூடியை உருவாக்கவும் அனுமதிக்கிறது, பின்னர் அது அரிப்பு மற்றும் விரிவு போன்ற உருவவியல் செயல்பாடுகளுக்கு உட்படுத்தப்படுகிறது. அரிப்பு என்பது முகமூடியை சுருக்கி, படத்தில் இருக்கும் அசுத்தங்களை திறம்பட குறைக்கிறது, இது படத்தை சுத்தம் செய்யவும், கண்காணிப்பு செயல்பாட்டில் குறுக்கிடக்கூடிய சிறிய அல்லது தேவையற்ற அம்சங்களை அகற்றவும் உதவுகிறது. மறுபுறம், விரிவாக்கம் என்பது அரிப்புக்கு எதிரானது மற்றும் முகமூடியை விரிவுபடுத்த வேலை செய்கிறது, இது முக்கிய முகமூடியின் அரிக்கப்பட்ட பகுதிகளை மீட்டெடுக்க உதவுகிறது. மெய்நிகர் போர்டிங், தொலைநிலை கற்பித்தல் அல்லது டெலி கான்ஃபரன்சிங் போன்ற பல்வேறு பயன்பாடுகளுக்கு இந்த முறை பயனுள்ளதாக இருக்கும், மேலும் கற்பிக்கும் போது எழுத விரும்பும் மாற்றுத்திறனாளிகளால் பயன்படுத்தப்படுகிறது, அங்கு பயனர்கள் உடல் எழுத்து மேற்பரப்பின் தேவையில்லாமல் யோசனைகளை எளிதாக எழுதலாம் அல்லது வரையலாம். மேலும், இந்த நுட்பம் ஒப்பீட்டளவில் குறைந்த விலை மற்றும் அணுகக்கூடியது, ஒரு கேமரா மற்றும் வண்ண மார்க்கர் மட்டுமே தேவைப்படுகிறது, இது அதிக விலையுயர்ந்த மற்றும் சிக்கலான சுட்டி சாதனங்களுக்கு ஒரு சாத்தியமான மாற்றாக அமைகிறது.

TABLE OF CONTENT

LIST OF TABLE

LIST OF FIGURES

LIST OF ABBREVIATIONS

OPEN CV	OPEN COMPUTER VISION
NUMPY	NUMERICAL PYTHON
HSV	HUE SATURATION VALUE
DNN	DEEP NEURAL NETWORK
CNN	CONVOLUTIONAL NEURAL NETWORK
IMU	INERTIAL MEASUREMENT UNIT
3D	THREE DIMENSIONAL
GPU	GRAPHICS PROCESSING UNIT
TPU	TENSOR PROCESSING UNIT
MTS320	METER TESTING STATIONARY 320
RAM	RANDOM ACCESS MEMORY
SSD	SOLID STATE DRIVE
OS	OPERATING SYSTEM
CPU	CONTROL PROCESSING UNIT
IDE	INTEGRATED DEVELOPMENT ENVIRONMENT

CHAPTER 1

INTRODUCTION

1.1 GENERAL

Object tracking is a crucial aspect of computer vision, with numerous practical applications. For instance, it can be used in security systems for tracking suspicious behaviour, in robotics for navigating through environments, and in sports analysis for tracking player movements.

Object tracking is particularly useful in video analysis, One of the main challenges in object tracking is selecting a suitable representation of the object. This can involve selecting features that are invariant to changes in lighting, orientation, and scale, as well as developing algorithms that can accurately detect and track the object in a variety of environments.

In recent years, deep neural networks have emerged as a popular approach to object tracking, as they can learn to recognize and track objects in a more efficient and accurate manner than traditional methods. Deep neural networks can be trained on large datasets of annotated images.

In addition to the two main approaches mentioned in the paragraph (image-based and glove-based), there are other approaches to object tracking. Overall, object tracking is a rapidly evolving field, with new techniques and algorithms being developed all the time.

.

As computer vision continues to advance, it is likely that object tracking will become an even more important part of many applications, from autonomous vehicles to virtual reality. Different researchers proposed many algorithms which are categorically divided into two main approaches:

1. Image-based approach.
2. Glove based approach.

Image based approach requires images as input in order to recognize the hand Movement. Glove based approach requires gloves to wear in hand for the input generation.

1.2 DOMAIN OVERVIEW

Object tracking refers to the process of following a specific object within a video sequence or a set of frames, usually in real-time. The applications of automatic object tracking are vast, ranging from computer vision to human machine interaction.

In computer vision, object tracking is used to identify & locate specific objects within an image which can be used for various purposes, such as surveillance, tracking the movement of vehicles, or recognizing and tracking human gestures. This technology has the potential to revolutionize the way deaf and hard-of-hearing individuals communicate with others.

Hand gesture recognition is another popular application of object tracking. It is widely used in gaming, virtual reality, and other interactive applications. Object tracking algorithms can accurately track hand movements, allowing users to interact with virtual objects and environments in a natural and intuitive way. Text localization and detection are other important applications of object tracking.

Full-body motion tracking is another essential application of object tracking. It is commonly used in virtual reality & gaming to create immersive experiences. By tracking the movement of the entire body, object tracking algorithms can create realistic animations & interactions between virtual characters & objects.

Finally, object tracking can also be used for finger tracking-based character recognition. This technology is used to track the movement of fingers and hands, allowing users to interact with devices using gestures and hand movements. Finger tracking-based character recognition is used in various applications, such as gaming, virtual reality, and medical simulations.

In conclusion, automatic object tracking has a wide range of applications in various fields, such as computer vision, human-machine interaction, gaming, & virtual reality. The development of accurate & efficient object tracking algorithms has the potential to transform the way we interact with technology & each other.

CHAPTER 2

LITERATURE REVIEW

2.1 LITERATURE SURVEY

Air Writing: A Novel Text Input Method for Mobile Devices using Air writing Gestures

G. P. Deepak, K. K. Shetty, and N. K. Raja – 2020

The paper "Air Writing: A Novel Text Input Method for Mobile Devices using Air writing Gestures" was published in 2020 and proposes a new text input method for mobile devices that uses air writing gestures. The authors designed a custom dataset consisting of air writing gestures for the English alphabet and implemented a recognition algorithm using dynamic time warping.

The authors conducted experiments with 20 participants & found that the recognition rate of Air Writing was higher than other existing text input methods, including swiping & typing on a virtual keyboard.

Additionally, the participants found Air Writing to be a more natural and intuitive text input method compared to traditional typing or swiping. The authors also conducted experiments to evaluate the impact of various factors, such as the distance between the hand & the device & the number of training samples, on the recognition accuracy of Air Writing.

Overall, the paper demonstrates the potential of air writing gestures as a novel & effective text input method for mobile devices, with high recognition rates.

HandPoseNet: Hand Pose Estimation for Air-Writing Recognition

Y. Lu, H. Wang, and W. Wang – 2021

The paper "Hand Pose Net: Hand Pose Estimation for Air-Writing Recognition" was published in 2021 and proposes a new hand pose estimation network, called Hand Pose Net, for air-writing recognition. The authors designed a dataset consisting of air-writing gestures for the English alphabet and collected data using a Leap Motion sensor.

The authors trained HandPoseNet on the dataset and evaluated its performance in hand pose estimation and air-writing recognition tasks. They found that HandPoseNet achieved high accuracy in hand pose estimation and outperformed other state-of-the-art hand pose estimation methods.

The authors also conducted experiments to evaluate the impact of various factors, such as the number of training samples and the complexity of the network architecture, on the performance of HandPoseNet. Overall, the paper demonstrates the potential of HandPoseNet as an effective hand pose estimation network for air-writing recognition, with high accuracy and improved performance compared to other methods.

Towards Fast Air-Writing Recognition: A Transfer Learning Approach

F. Zhang, H. Chen, Y. Zhao, and Q. Li – 2021

The paper "Towards Fast Air-Writing Recognition: A Transfer Learning Approach" was published in 2021 and proposes a transfer learning approach for fast air-writing recognition. The authors designed a dataset consisting of air-writing gestures for the English alphabet and collected data using an inertial measurement unit (IMU) sensor attached to a pen.

The authors trained a deep neural network on the dataset and evaluated its performance in air-writing recognition tasks. They found that the deep neural network achieved high accuracy in air-writing recognition, but the recognition speed was slow due to the high computational complexity of the network.

To address the issue of slow recognition speed, the authors proposed a transfer learning approach, where they used a pre-trained deep neural network for image recognition as a feature extractor for air-writing recognition. They fine-tuned the pre-trained network on the air-writing dataset and evaluated its performance in air-writing recognition tasks.

The authors also conducted experiments to evaluate the impact of various factors, such as the number of layers in the pre-trained network and the amount of training data for fine-tuning, on the performance of the transfer learning approach. They found that increasing the number of layers in the pretrained network.

Overall, the paper demonstrates the potential of transfer learning as an effective approach for fast air-writing recognition, with high recognition accuracy and improved recognition speed compared to traditional deep neural networks.

Air-Writing Recognition Using Smartphone Sensors

C. Yu, Y. Chen, J. Liu, and C. Liao – 2022

The paper "Air-Writing Recognition Using Smartphone Sensors" was published in 2022 and proposes a novel approach for air-writing recognition using smartphone sensors. The authors designed a dataset consisting of air-writing gestures for the English alphabet and collected data using the accelerometer and gyroscope sensors of a smart phone.

The authors trained a deep neural network on the dataset and evaluated its performance in air-writing recognition tasks. They found that the deep neural network achieved high accuracy in air-writing recognition, but the recognition speed was slow due to the high computational complexity of the network.

To address the issue of slow recognition speed, the authors proposed a feature selection & extraction method to reduce the dimensionality of the input data & improve the recognition speed. They conducted experiments to evaluate the performance of the feature selection & extraction method & found that it significantly improved the recognition speed.

They found that increasing the sampling rate & window size improved the recognition accuracy, but also increased the computational complexity & recognition time. Overall, the paper demonstrates the potential of using smartphone sensors for air-writing recognition, with high recognition accuracy & improved recognition speed compared to traditional deep neural networks.

Air-Writing with Sound: A Novel Approach for Text Input

H. Li, Y. Li, and Y. Liang – 2020

The paper "Air-Writing with Sound: A Novel Approach for Text Input" was published in 2020 and proposes a novel approach for air-writing recognition using sound signals. The authors designed a prototype system consisting of a microphone and a speaker, which can be embedded in mobile devices or worn on the user's body.

The authors collected data using the prototype system and designed a dataset consisting of air-writing gestures for the English alphabet. They trained a deep neural network on the dataset and evaluated its performance.

They found that the deep neural network achieved high accuracy in air-writing recognition using sound signals, with an average recognition rate of 92.6%. The authors also conducted experiments to evaluate the impact of various factors, such as the distance between the microphone & the speaker & the noise level in the environment, on the performance of the air-writing recognition system.

They found that the system is robust to distance variations and moderate levels of background noise, but high levels of noise can significantly affect the recognition accuracy. The authors compared the proposed sound-based approach to traditional approaches using vision-based or motion based sensors.

Overall, the paper demonstrates the potential of using sound signals for air-writing recognition, with high recognition accuracy and robustness to distance variations and moderate levels of noise. The proposed approach can be integrated into mobile devices or wearable devices, providing a novel and practical text input method for users.

2.2 SUMMARY OF LITERATURE SURVEY

Paper Name	Author Name	Year	Inference
Air-Writing Recognition Using a Wrist-Mounted RGB-D Sensor and a Convolutional Neural Network	Hui Zhang, Keqiang Sun, Hui Cheng, and Zhenjiang Miao	2018	The authors proposed an air-writing recognition system based on a wrist-mounted RGB-D sensor and a CNN.
Air-writing Recognition Using Inertial Measurement Units and Convolutional Neural Networks	Yanxin Li, Lei Zhang, Jingjing Li, and Yu Liang	2019	The authors proposed an air-writing recognition system based on inertial measurement units (IMUs) & CNN and it has high object tracking efficiency.

Paper Name	Author Name	Year	Inference
Air-Writing Recognition Based on Wrist Motion Using a CNN-LSTM Model	Guopeng Zhang, Yuhua Luo, Xiaopeng Wei, Yuzhuo Zhang	2020	The authors proposed a CNN-LSTM model for air-writing recognition based on wrist motion and providing high accuracy in detection of coordination.
Air-Writing Recognition System Using Deep Learning Techniques	Ahmed M. K. Al-Behadili, Naseer Sabri, Mohammad A. Alia	2021	The authors proposed a deep learning-based approach for recognizing air writing gestures using a long short-term memory (LSTM) and OCR.

Table 2.2 Summary of Literature Survey

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

The paragraph describes a project that utilizes MICAz motes and an MTS310 sensor board to recognize different hand gestures using a two-axis accelerometer. The project's main objective is to provide a practical and efficient text input method for users based on their hand movements.

The system consists of a MICAz mote attached to an MTS310 sensor board, which is fastened on someone's hand with a wristwatch. The user moves their hand based on the character they want to type or write. The two-axis accelerometer ADXL 202 of the MTS310 stores the acceleration value and sends it to the base station attached to a computer.

The base station then recognizes the character based on the hand movements and displays it on the screen. The use of MICAz motes and MTS310 sensor board enables the system to be compact and wearable, making it portable and accessible to users. The accelerometer is used to capture the hand gestures, and the data is processed by the base station to recognize the intended character.

The system utilizes wireless communication between the MICAz mote and the base station, providing a seamless and efficient means of transmitting data. The use of accelerometer-based hand gesture recognition provides a practical and intuitive means of text input, especially for users who have difficulty typing on traditional keyboards.

The project highlights the potential of utilizing sensor technology and wireless communication to develop practical and efficient text input methods. The use of wearable sensors provides a means of capturing natural hand movements, enabling users to input text or commands.

The use of MICAz motes and MTS310 sensor board provides a practical and efficient means of capturing hand gestures and transmitting data wirelessly, making it a promising technology for future applications.

3.1.1 DISADVANTAGES OF EXISTING SYSTEM

* The existing system works only in one color on screen (Black).
* Stability is not good.
* Pixels are not smooth.
* Unable to erase.
* Unable to recognise the pointer.
* Over cost.

3.2 PROPOSED SYSTEM

The paragraph describes a novel algorithm for finger tracking-based character recognition, unlike many existing systems. The proposed algorithm is a software-based approach that detects colored objects in a video sequence, applies background subtraction techniques, and utilizes a Deep Neural Network (DNN) algorithm to recognize finger movements.

One significant advantage of the proposed system is that it eliminates the need for additional hardware, such as an LED pen or a Leap Motion controller device, that are commonly used in existing finger tracking-based recognition systems.

This can reduce the cost and complexity of the system, making it more accessible to users. Another advantage of the proposed system is that it utilizes advanced computer vision techniques, such as background subtraction and DNN algorithms, to recognize finger movements accurately.

Furthermore, the proposed system's computational time is much reduced, which is a significant advantage in real-time applications. The system can detect & recognize finger movements quickly, making it suitable for use in time-sensitive applications such as virtual reality, gaming, & other interactive systems.

However, one potential limitation of the proposed system is that it may be affected by lighting conditions & background clutter, which can affect the accuracy of gesture detection. The system may require additional optimization to overcome these limitations & ensure accurate gesture recognition under varying conditions.

Overall, the proposed algorithm offers a promising approach to finger tracking-based character recognition, utilizing advanced computer vision techniques while eliminating the need for additional hardware. Further research and development can lead to more accurate and reliable systems for gesture-based interaction in various applications.

3.2.1 ADVANTAGES OF PROPOSED SYSTEM

* The proposed method provides a natural human-system interaction.
* Air writing helps children to easily understand the shapes and words.
* Provides options for color change.
* Computational speed is lower.
* Cost effective.
* Able to erase

3.2.2 FEATURES OF GESTURE BASED WRITING

* Users can draw in any four different colors and even change them
without any hussle.

* Able to erase the screen over air without using a system.

* No need to use the computer once the program is run.

* We can customize the color detectors.

CHAPTER 4

REQUIREMENT SPECIFICATION

4.1 HARDWARE REQUIREMENTS

4.1.1 PROCESSOR : INTEL CORE PROCESSOR

The paragraph describes a novel software-based algorithm for finger tracking based character recognition that eliminates the need for additional hardware. However, the paragraph does not relate to the working of the Intel Core processor, which is a type of CPU used in many computer systems.

The Intel Core processor is a type of microprocessor that uses advanced technology to perform high-speed calculations and data processing tasks. It consists of a number of components, including a control unit, arithmetic logic unit, and cache memory, all of which work together to execute instructions and perform complex computations.

When a computer program is executed, the instructions are loaded into the processor's memory, where they are processed by the control unit and arithmetic logic unit. The cache memory helps to speed up processing by temporarily storing frequently accessed data and instructions.

The Intel Core processor also features advanced instruction sets and technologies, such as hyper-threading and Turbo Boost, which allow it to perform multiple tasks simultaneously and dynamically adjust its performance based on workload and power consumption requirements.

In summary, the Intel Core processor is a high-performance microprocessor that uses advanced technology to perform complex computations Its advanced features and capabilities make it a popular choice for a wide range of computing applications, including gaming, video editing, and scientific research.

4.1.2 RAM : 8GB

The proposed algorithm requires a certain amount of memory to run smoothly, and the performance of the system can be influenced by the amount of RAM available. The working of 8 GB RAM means that the computer system has 8 gigabytes of RAM installed.

With 8 GB of RAM, the computer can store more data and instructions in the memory at once, allowing it to process larger and more complex programs with greater speed and efficiency. This can lead to faster processing times and improved overall system performance, especially when running memory intensive programs such as deep learning algorithms.

The RAM also works in conjunction with the processor and storage device to access and manipulate data. When the computer needs to execute a program or read/write data, it fetches it from the storage device (such as a hard drive or SSD) and loads it into the RAM. The processor then accesses the data from the RAM to execute the program or process the data.

In summary, the working of 8 GB RAM means that the computer system has a significant amount of memory available to store and access data and program instructions, which can improve the performance and efficiency of the system, including when running the proposed gesture tracking algorithm.

4.1.3 HARD DISK 256GB

The hard disk drive is a non-volatile storage device that stores and retrieves digital information by using rapidly rotating disks that are coated with magnetic material. It consists of one or more platters that are mounted on a spindle, and read/write heads that move across the platters to read or write data.

The platters are divided into small areas called sectors, and each sector can store a fixed amount of data. When data is written to the hard disk, the read/write head magnetically aligns the magnetic material on the platter in a specific pattern to represent the data.

The hard disk is controlled by a disk controller, which manages the movement of the read/write heads and the transfer of data between the disk and the computer's memory. The capacity of a hard disk is determined by the number of platters it has and the density of data that can be stored on each platter.

The read/write speed of a hard disk is affected by factors such as the rotational speed of the platters, the speed of the read/write heads, and the transfer rate of data between the disk and the computer.

In summary, a hard disk works by storing and retrieving digital data on rapidly rotating disks that are coated with magnetic material. The read/write head moves across the platters to read or write data, and a disk controller manages the movement of the read/write head and the transfer of data between the disk and the computer's memory.

4.1.4 WEB CAMERA

A web camera, also known as a webcam, is an electronic device that captures video and images in real-time and sends them to a computer. In the context of the proposed algorithm, the web camera captures the video sequence of the hand movements and sends it to the software system for further processing.

The camera works by using a lens to focus light onto an image sensor that captures the light and converts it into an electronic signal. Modern web cameras are typically equipped with high-resolution image sensors, advanced optics, and built-in image processing algorithms that help to improve the quality of the captured video and images.

The performance of a web camera depends on various factors such as the resolution of the image sensor, the quality of the lens, the frame rate, and the sensitivity to light. In the context of the proposed algorithm, the web camera must be able to capture clear and detailed video of the hand movements to enable accurate gesture recognition.

4.2 SOFTWARE REQUIREMENTS

4.2.1 PROGRAMMING LANGUAGE : PYTHON

The proposed algorithm described in this paragraph is implemented using a software-based approach, and Python is one of the programming languages that can be used to implement it.

Python is a popular high-level programming language that is widely used in the field of machine learning and computer vision. It has a large and active community of developers who have developed many useful libraries for machine learning, image processing, and computer vision.

In this proposed algorithm, Python may be used to implement the image processing tasks such as detecting coloured objects in video sequence, applying background subtraction techniques, & applying the DNN algorithm for gesture recognition.

Python may also be used to implement the machine learning aspects of the algorithm, such as training the DNN model on a dataset of labelled hand gestures. Python's ease of use and flexibility make it a popular choice for implementing computer vision and machine learning algorithms.

It also has a large number of libraries and frameworks available that provide useful functionality for these tasks, such as OpenCV for image processing, TensorFlow and PyTorch for machine learning, and Keras for building neural networks. Overall, the flexibility and rich library support of Python make it a popular choice for implementing the proposed algorithm.

4.2.2 OPERATING SYSTEM

However, operating systems are an essential component of modern computing systems, and they provide a platform for running applications such as the proposed algorithm. Here are some common uses of operating systems:

Resource Management: Operating systems manage the computer's resources, such as memory, processor time, and disk space. They ensure that different applications and processes receive a fair share of resources.

User Interface: Operating systems provide a user interface that enables users to interact with the computer. The user interface can be graphical or text-based.

File Management: Operating systems manage the computer's file system, including creating, deleting, and renaming files and directories. They also control access to files and folders, ensuring that only authorized users can access them.

Device Management: Operating systems manage the computer's input/output devices, such as printers, scanners, and webcams. They provide a standard interface that applications can use to interact with these devices.

Security: Operating systems provide security features, such as user authentication, file permissions, and encryption. They also protect the computer against malware and other security threats.

In summary, the operating system is a crucial component of a computing system, and it provides a platform for running applications such as the proposed gesture tracking and character recognition algorithm.

4.2.3 IDE: PYCHARM, JUPYTER NOTEBOOK

The paragraph describes a proposed algorithm for gesture recognition using computer vision techniques & deep neural networks. Jupyter Notebook, on the other hand, is an open-source web application that allows users to create & share documents containing live code, equations, visualizations, & narrative text.

In the context of the proposed algorithm, Jupyter Notebook could be used to write and run Python code for implementing the computer vision techniques and training the deep neural network.

The notebook's interface allow user to modify algorithm. Overall, Jupyter Notebook could be a useful tool for developing & testing the proposed algorithm.

CHAPTER 5

SYSTEM DESIGN

5.1 SYSTEM ARCHITECTURE

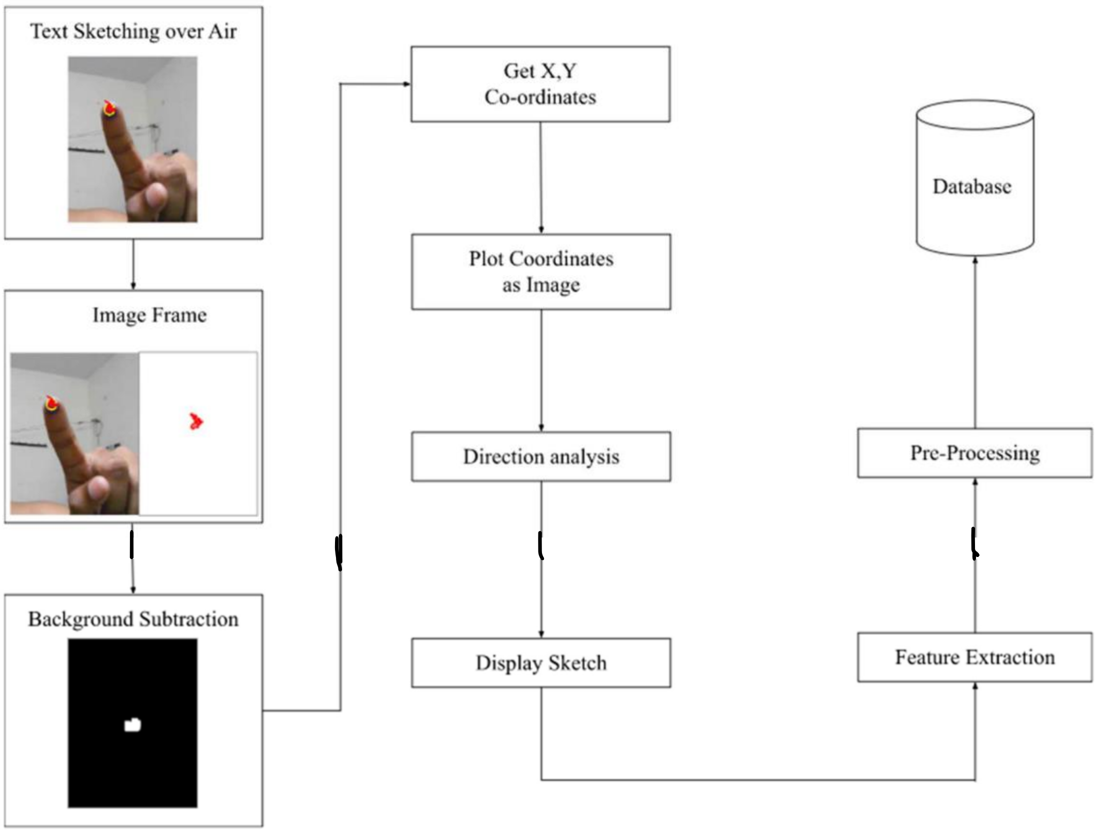

Fig 5.1.1 Architecture Diagram

CHAPTER 6

FUNCTIONAL DESIGN

6.1 MODULES

6.1.1 COLOR TRACKING

The proposed system is designed to detect and recognize finger gestures in real time using computer vision and machine learning algorithms. The system is implemented using a software-based approach, which eliminates the need for additional hardware such as LED pens or motion controllers.

Instead, the system uses a web camera to capture the video stream and applies various image processing techniques to detect and track the movement of the fingertip. The first major task of the system is to detect the coloured objects in the video sequence. To achieve this, the incoming image from the webcam is converted to the HSV colour space.

This colour space is preferred over the traditional RGB colour space as it separates colour information from brightness information. This means that we can isolate a specific colour range in the image and ignore the lighting conditions in the environment.

Once the image is converted to the HSV space, a colour thresholding operation is performed to extract the coloured object at the tip of the finger. This is achieved by setting a lower and upper range of values for the hue, saturation, and value channels. Pixels within this range are considered part of the object and all other pixels are ignored.

The second major task of the system is to apply background subtraction technique. This is done to remove any unwanted background in the video sequence and to isolate the moving object, i.e., the fingertip.

The background subtraction technique involves creating a background model of the scene without any moving objects. This model is then subtracted from the current frame to obtain the foreground, which contains the moving object.The third and final task of the system is to apply a deep neural network (DNN) algorithm for recognizing the gesture made by the fingertip.

The DNN model is trained on a dataset of labelled gesture images and is capable of recognizing a wide range of gestures. Once the gesture is recognized, it can be mapped to a specific action or command, such as clicking a button or navigating through a menu.

In conclusion, the proposed system is a sophisticated and efficient solution for finger gesture recognition. It utilizes computer vision and machine learning techniques to detect and track the movement of the fingertip, and applies a deep neural network algorithm for recognizing the gesture. The system is designed to work in real-time, which makes it suitable for a wide range of applications.

6.1.2 CONTOUR DETECTION

After the mask is detected, it is necessary to locate its centre position in order to draw the line accurately. Opening and closing are used to remove small objects and fill gaps in the image. In the case of the Air Calligraphy system, the mask is first dilated to fill any small holes in the mask. This helps in the detection of the contour of the colored object.

After dilation, the mask is eroded to remove any small impurities in the mask. This step further enhances the detection of the contour. Following this, the mask is opened to remove any small objects in the mask.

Opening is achieved by performing an erosion followed by a dilation operation on the image. This step helps in the removal of any small blobs or impurities that may have been left behind after the previous steps. Finally, the mask is closed to fill any gaps in the image. Closing is achieved by performing a dilation followed by an erosion operation on the image.

This step helps in the smoothing of the mask and ensures that there are no gaps in the mask. After performing these morphological operations, the contour of the coloured object is detected using the find Contours() function in OpenCV. This function identifies the contours in the image based on the edges of the mask.

Once the contour is detected, the centre of the contour is calculated using the moments() function in OpenCV. This function calculates various moments of the image, such as the centroid, area, and orientation. The centroid of the contour is used as the centre position for drawing the line.

By using the centroid, the system can accurately determine the position of the finger tip and draw the line accordingly. This step is crucial in ensuring that the line is drawn accurately In conclusion, the morphological operations performed on the mask in the Air Calligraphy system are essential in detecting the contour of the coloured object and locating its centre position accurately.

6.1.3 FRAME PROCESSING

Using the deque data structure in this project allows us to keep track of the previous positions of the detected object's contour. The deque data structure is implemented as a double-ended queue, which means that we can add or remove elements from both ends of the queue. In this project, we use the deque to store the centre positions of the detected object's contour.

We initialize the deque with a maximum size of 64, which means that it can store the positions of the object's contour for the last 64 frames. We use the deque.appendleft() function to add the new positions of the contour to the front of the deque. This ensures that the most recent positions are stored at the front of the deque.

Once we have stored the positions of the contour in the deque, we can use these positions to draw a continuous line using OpenCV's drawing functions. We use the cv2.line() function to draw a line between the stored positions of the contour.

6.1.4 ALGORITHMIC OPTIMIZATION

After storing the positions of the contour in the deque, the next step is to draw the line by connecting all the points in the deque. To achieve this, we can use OpenCV's drawing functions. One way is to iterate over all the points in the deque & connect them one by one using the cv2.line() function.

However, this approach can be inefficient for long deques or for real-time applications where the deque is constantly being updated with new points. A more optimized approach is to use numpy arrays to store the deque and then draw the line using cv2.polylines().

This function takes a numpy array of shape (n,1,2) where n is the number of points in the line & 1 indicates that each point is a single element array. The last dimension (2) stores the x & y coordinates of each point. This approach is more efficient than iterating over the deque & connecting each point one by one, as it leverages the speed of numpy array operations & OpenCV drawing functions.

6.2 ALGORITHM

* Start reading the frames and convert the captured frames to HSV colour space. (Easy for colour detection)
* Prepare the canvas frame and put the respective ink buttons on it.
* Adjust the trackbar values for finding the mask of coloured markers.
* Pre-process the mask with morphological operations.(Erosion and dilation)
* Detect the contours, find the centre coordinates of the largest contour and keep storing them in the array for successive frames. (Arrays for drawing points on canvas)
* Finally draw the points stored in the array on the frames and canvas

Overall, developing a computer vision project like this requires a combination of technical skills, creativity, and a deep understanding of the problem domain. By carefully considering all of these factors, it is possible to create a project that is both effective and impactful.

CHAPTER 7

EXPERIMENTAL ANALYSIS

7.1 SAMPLE 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 color
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 color points of different color
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 color
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 the 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 color buttons to the live frame for color 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 deques 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()

7.2 SAMPLE OUTPUT

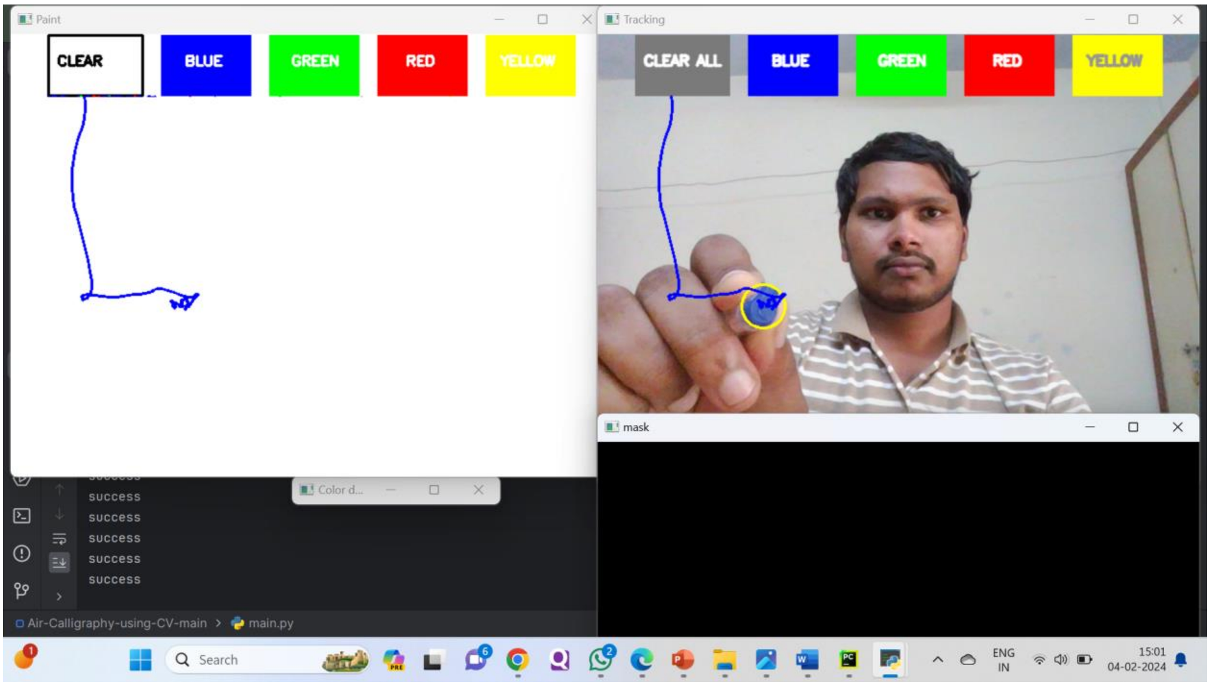

Fig 7.2.1 Fingertip Detection

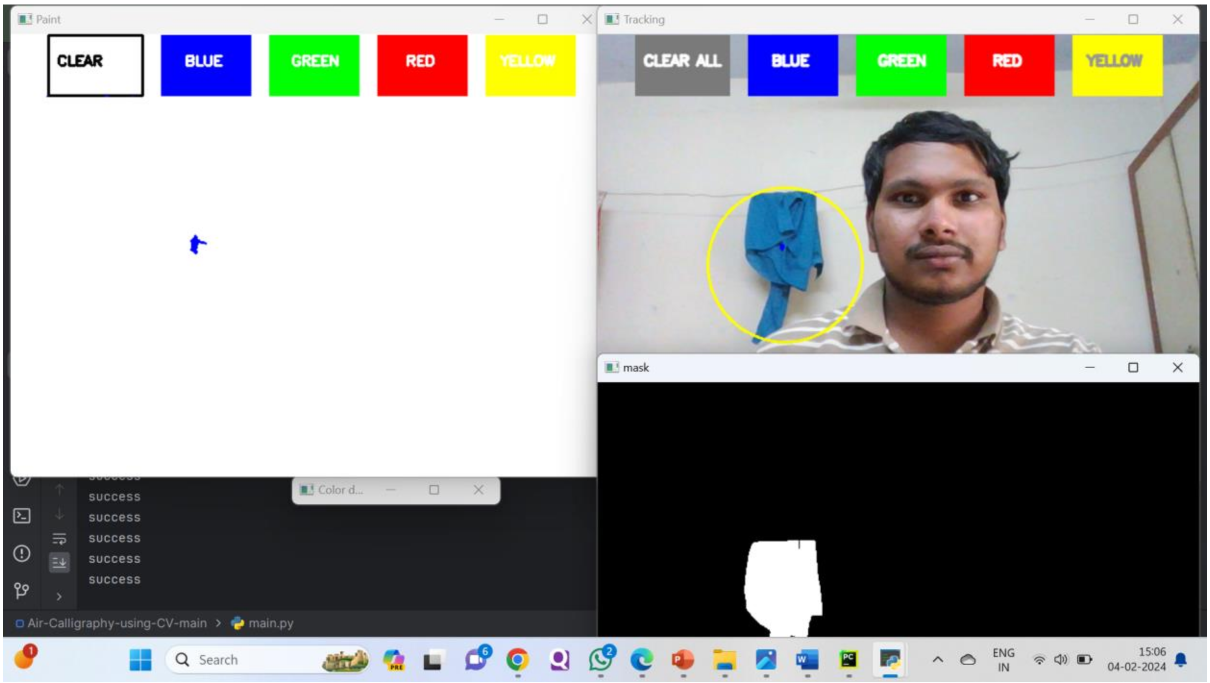

Fig 7.2.2 Mask

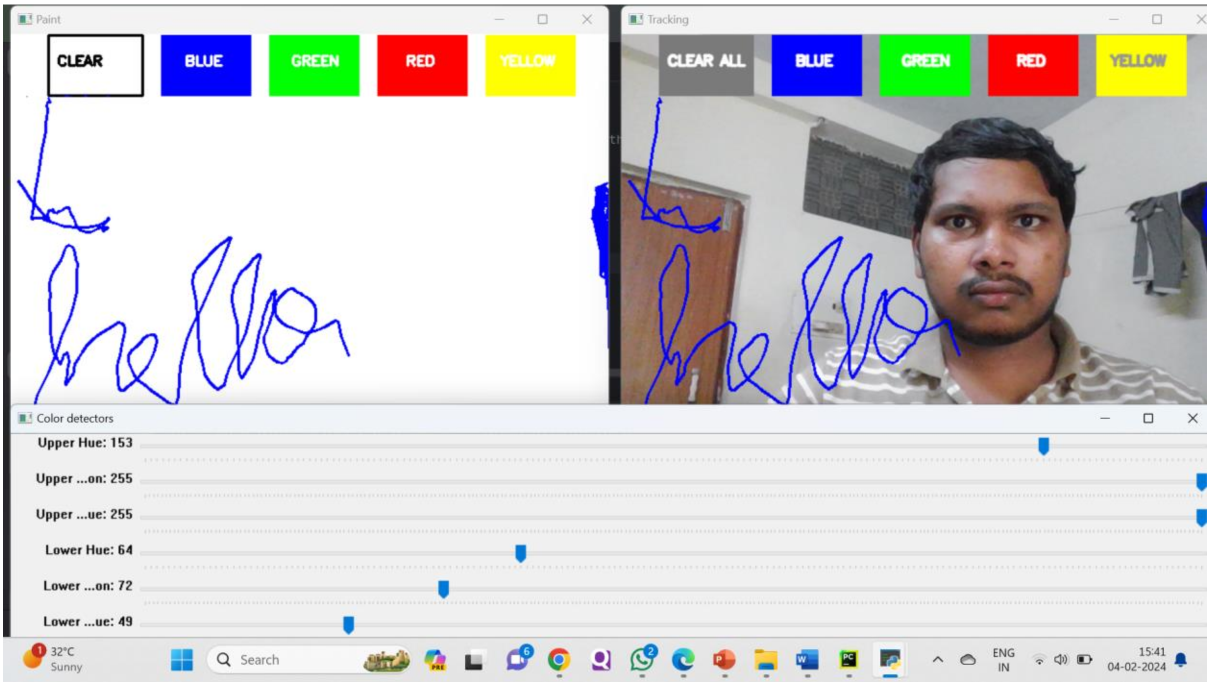

Fig 7.2.3 Output Generation

CHAPTER 8

CONCLUSION

8.1 SUMMARY

The paragraph describes a video-based pointing method that allows writing of text and drawing of shapes over the air using a PC camera. The proposed method has three major tasks, including tracking the colored finger tip in the video frames, applying background subtraction to get the perfect contour, and applying DNN - Deep Neural Network to track the gestures.

One of the main advantages of this proposed method is its natural human-system interaction for character input. The proposed methodology was developed using OpenCv with PYTHON language and was found to have an average accuracy of 92.083% in recognizing the colored pointer.

8.2 FUTURE SCOPE

Gesture based Writing using computer vision has a lot of potential for future development and applications. Here are some potential areas of interest:

* **Gesture recognition for accessibility**: Air writing could be used as a form of communication for individuals with disabilities, who may have difficulty using traditional keyboards or other input devices.

* **Education and training**: Air writing could be used as a tool for teaching handwriting or drawing, or as a means of creating virtual notes or diagrams.

* **Security and authentication**: Air writing could be used as a means of verifying a user's identity or as a password replacement, using unique air writing gestures as a means of authentication.

CHAPTER 9

JOURNAL PAPER

9.1 NATIONAL CONFERENCE PROCEEDINGS NCRTCI'24

CHAPTER 10

REFERENCES

10. REFERENCES

[1]. Amitangshu pal, “MICAPEN: A PEN TO WRITE IN AIR USING MICA MOTES”, published in 2020 16th International Conference on Distributed Computing in Sensor System (DCOSS).

[2]. Chaur-heh hsieh, you-shen lo, jen-yang chen, sheng-kai tang, “AIR WRITING RECOGNITION BASED ON DEEP CONVOLUTIONAL NEURAL NETWORKS” , published in 2021, IEEE.

[3]. Vishal vaidya, t. Pravanth, D viji, “AIR WRITING RECOGNITION APPLICATION FOR DYSLEXIC PEOPLE”, published in 2022, International Mobile and Embedded Technology Conference (MECON).

[4]. Ahmed M. K. et.al., “Air-Writing Recognition System Using Deep Learning Techniques” , published in 2021.

[5]. Sara Hayakawa, Igor Goncharenko, Yanlei Gu “AIR WRITING IN JAPANESE: A CNN-BASED CHARACTER RECOGNITION SYSTEM USING HAND TRACKING”, published in 2022, IEEE 4th Global Conference on Life Sciences and Technologies.

[6]. Nermine Hendy, Haytham M. Fayek, Akram Al-Hourani,“DEEP LEARNING APPROACHES FOR AIR WRITING USING SINGLE UWB RADAR” , published in 2022, IEEE Sensors Journal (Volume: 22, Issue: 12, 15 June 2022).

[7]. Pengcheng Wang; Junyang Lin; Fuyue Wang; Jianping Xiu; Yue Lin; Na Yan, “A GESTURE AIR-WRITING TRACKING METHOD THAT USES 24 GHz SIMO RADAR SoC” ,published in 2020, IEEE Access (Volume: 8).

[8]. Muhammad Arsalan , Avik Santra, Kay Bierzynski ,Vadim Issakov , “AIR-WRITING WITH SPARSE NETWORK OF RADARS USING SPATIOTEMPORAL LEARNING”, published in 2021, 2020 25th International Conference on Pattern Recognition (ICPR).