AIR WRITING AND RECOGNITION SYSTEM

Done at

CODEBUZ, KOTTAYI

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF

BACHELOR OF SCIENCE IN COMPUTER SCIENCE

OF

UNIVERSITY OF CALICUT

SUBMITTED BY

ABIN P (VVAVSCS005)

SHREYAS S (VVAVSCS019)

NANDHANA S (VVAVSCS027)

NITHEESH G (VVAVSCS034)

UNDER THE GUIDANCE OF

Mrs. MEENA P S

(ASSISTANT PROFESSOR OF COMPUTER SCIENCE - V.V COLLEGE)

PG DEPARTMENT OF COMPUTER SCIENCE

V.V. COLLEGE OF SCIENCE & TECHNOLOGY

(AFFLIATED TO UNIVERSITY OF CALICUT)

KANJIKODE, PALAKKAD – 678 621

ACCREDITED BY NAAC WITH B+

MARCH 2024

V.V COLLEGE OF SCIENCE & TECHNOLOGY

KANJIKODE, PALAKKAD – 678 621.

(Affiliated to Calicut University)

ACCREDITED BY NAAC WITH B+ GRADE

CERTIFICATE

This is to certify that the project work entitled "AIR WRITING AND RECOGNITION SYSTEM" *done at* **CODEBUZ** *is a bonafide record of the work done by* Mr./ Ms. ***ABIN P (VVAVSCS005), SHREYAS S (VVAVSCS019), NANDHANA S (VVAVSCS027), NITHEESH G (VVAVSCS34)*** *submitted in partial fulfillment of requirements for the award of the degree of bachelor of science in computer science of University of Calicut.*

Viva – voice examination held on______________________

PRINCIPAL H.O.D INTERNAL GUIDE EXTERNAL EXAMINERS

1.

2.

ACKNOWLEDGMENT

We have great pleasure in acknowledging our sincere gratitude to all who have been given the helping hands in the successful project. First, we are all ways thankful to the Almighty GOD who have showered blessings on us & gave us strength for doing the project. We would like to owe our sincere thanks to the Principal: Dr V. K. SUDHAKARAN M.com,M.phil,Ph.D(Principal), VV College of Science and Technology, Pudussery Central, Kerala for providing us the best facilities and atmosphere according to our interest for the successful completion & presentation of our project. We are most likely to give our sincere gratitude to Mrs. SURABHILA.K MCA,M.Phil (HOD Computer Science Dept) and our internal Project guide Mrs. MEENA P S MCA,.M.Phil (ASST professor Computer Science Dept) for all valuable advice & support given to us. Finally, we express our thanks to all our friends whose valuable suggestions, criticism and encouragements that helped to make this project a great success. We also thank our parents for their prayer through which our project was successfully completed

SYNOPSIS

Writing in air has been one of the most entrancing and challenging research regions in field of image processing and pattern recognition in the recent years. It contributes enormously to the progression of an automation process and can get to the next level the connection point among man and machine in various applications. Object following is considered as a significant task within the field of Computer Vision. The development of quicker PCs, accessibility of cheap and great quality camcorders and requests of computerized video investigation has given prevalence to object tracking strategies.

The Air Writing and Recognition System is an innovative project that aims to bridge the gap between human-computer interaction and natural language processing. Utilizing advanced Computer Vision techniques and Deep Learning algorithms, this system is designed to interpret and translate air-written text into digital characters. The project leverages a convolutional neural network (CNN) to process visual input from a standard webcam, which captures the user's hand movements as they write characters in the air.

The primary objective of this project is to develop a robust system capable of recognizing air-written text with high accuracy and speed. This system aims to provide a seamless and intuitive way for users to input text without the need for physical hardware, such as keyboards or trackpads. It has potential applications in virtual reality environments, assistive technologies for individuals with disabilities, and as a tool for language learning.

The Air writing recognition system uses the webcam of a computer to track character and digits written in the air by the user, Computer then uses a mask to track a specific color selected by the user. This mask then tracks the color and it is then drawn onto a canvas (i.e. plain whiteboard). We use this canvas image as an input for the recognition

model to understand the air written words to the character or digits into one of 62 different classes (i.e. 10 integers, 26 Capital letters, 26 small letters).

The system will have two modes of operation, selectable by the user: ‘Air Write’ and ‘Write & Recognition’:

Air Write : In this mode, the system will allow the user to write characters in the air, which will be displayed on the screen in real-time.

Write & Recognition : In this mode, the system will not only display the characters written in the air but also recognize and interpret them.

The frontend of the project is **TKinter**, **Python** and the backend is **Dataset** which makes it easy for creating and generating code. **Windows** is used as Operating System.

Modules:

- Finger Recognition
- Writing With Free Hand
- Tools Integration
- Erase Virtually
- Air Writing Recognition

CONTENTS

1. INTRODUCTION

1.1 PROJECT OBJECTIVE

All we want in this undertaking is a PC with a web camera introduced in it. We will prepare our PC or our screen to peruse anything that the client will compose infront of the screen. We have involved Open CV for object identification that is with which the client will be writing on air and we have used Python language for coding. First and foremost, after code is executed, we get a white screen shown in the screen and anything we write on air Infront of camera it tracks the article whose properties we have proclaimed in the code. After that tracked points are connected and projected on the screen .

In the period of digital world, traditional art of composing is being supplanted by advanced digital art . Digital art refers to forms of expression of art form with digital form. Depending on present day science and innovation is the distinctive characteristics of the digital manifestation. Digital Art incorporates numerous methods of writing like by utilizing Keyboard, Touch screen surface, Digital pen, pointer, utilizing electronic hand gloves and so forth. Traditional Art refers to the fine art which is made before the digital art. The Traditional way incorporates pen and paper, chalk also, board for writing. Digital Art and Traditional art are interrelated and interdependent .Social improvement is certainly not a people's will, yet at the same the at any rate, requirements of human existence are the vitally main impetus. The same circumstance occurs in Art. In the current conditions, Digital Art and Ttraditional Art are comprehensive of the advantageous state, so we really want to efficiently comprehend the essential knowledge on the structure between digital art and traditional art .The fundamental point of advanced workmanship is of building hand signal acknowledgment framework to write digitally.

Handwriting analysis could be a classic, well explored downside within the realm of introductory Machine Learning that encompasses several necessary topics of neural networks. Air Writing is particularly helpful for user inter phases that don't permit the user to depend on a keyboard, a pen a trackpad / touch screen, or for test input for sensible system management, among several applications. Air Writing differs from typical handwriting; the one that performs Air Writing will solely use associate degree unreal co-ordinate to guide the writing motion. The variability of motion knowledge represents a letter is so significantly broader in Air Writing than in Paper writing. Our project aims to use a mixture of computer vision and hand writing recognition to form a system that acts as a virtual white board. Model users would be ready to write "air words" facing a web camera either real-time or in advance and have those gestures translated into letters or digits. The aim of the project is to further explore the task of classifying hand written text and to convert hand written texts into a digital format and achieving a virtual white board system at a cost that is accessible to the average user. We want to introduce alternative interfaces for communication that have high affordability, usability, and accessibility.

Writing in Air is a Fascinating thing. This can be achieved by Computer Vision Techniques . Air Canvas is a novel way of painting virtually without using a physical brush or canvas. It is made of system that can catch movements of artists and can draw even without touching keyboard, mouse, or touchpad. Here fingertip detection and tracking using MediaPipe are used to achieve the objective. The colour marker is detected on index finger and Object tracking is considered an important task within the field of Computer Vision. Hand Tracking and Shapes Drawing is done using Computer Vision frameworks in python.

Air-writing is useful for users who don't want to type on a keyboard or write on a track pad/touch screen, as well as for text input for smart device control and a variety of other uses. Air writing intrigue many medical and engineering felids such as neurological department, physiatrist department and many other medical felids and engineering felids such as IoT and use cases in smart home handsfree appliances.

1.2 COMPANY PROFILE

Codebuz is a software development startup company that specializes in web apps. We do know how to get it all done. So, feel free to call us. At Codebuz, our mission is to develop and deliver technology solutions that change the way your business operates; at a competitive budget, with market leading technology and with a quick turnaround. As a customer, you are always in control and fully updated on the project status, and your feedback is critical to the continuous improvement.

2. SYSTEM ANALYSIS

2.1 INITIAL INVESTIGATION

In this the expresses that they have involved open CV for Object Detection that is with which the client will write on air and we have utilized Python language for coding since python is a simple language and easily understandable . After code is executed, we get a white screen shown in the Monitor and anything that we write on air Infront of camera it tracks the article whose properties we have declared in the code after that tracked points are associated and anticipated on the screen.

We use python as our essential language and its packages OpenCV and NumPy. Basically, OpenCV is a library of programming functions mainly aimed at PC vision. Initially created by Intel, it was later upheld by Willow Garage. And NumPy is basically, a library for the Python programming language, adding support for enormous, multi-dimensional arrays and lattices, along with an enormous collection of high-level numerical capacities to work on these arrays. The Project was fundamentally a PC vision application which utilizes the webcam of your gadget. By opening the webcam of your gadget all you want is to simply hold the pencil or finger tip in our grasp, then, at that point, drag the finger or pencil on air in front of your gadget camera.

In this task we are playing out the morphological tasks which are a bunch of tasks that processes pictures based on shapes. These apply a structuring component to an info picture and produce a output Image. Developing a connection point between human hand and the framework utilizing opencv strategies and python language to pick the colour and draw utilizing hand on the created drawing region. This framework utilizes a camera device also for Computer Vision Software to follow the way of our Finger Tip . This powerful specialized technique decreases mobile and PC usage by taking out the need to write.

2.2 EXISTING SYSTEM

Generally, in the field of 3D vision systems, the advanced techniques such as Leap motion and Kinect Cameras are used for writing the text in air. Tracking systems such as HTC Vive virtual reality (VR) had incorporated this mechanism but the system comes at a high cost that also requires an addition extensive (expensive) tracking system. Air-Writing allows for an entirely new text input interface that does not require much more than the computer itself. Our proposed method is done very easily by everyone and can be incorporated into their software. Here air-writing characters are recognised by using a library called OpenCV, that lets us detect the object we want to write with and then track its path. Any object like pen, pencil or a fingertip can be tracked by our system. We display to the user the character he/she has written in air by tracing the path of the object that is used to write in air.

Limitations of Existing System

- We waste a lot of paper in scribbling, composing, drawing, and so on.
- A few essential realities incorporate - 5 liters of water on normal are expected to make one A4 size paper, 93% of composing is from trees, half of business waste is paper, 25% landfill is paper, and the rundown goes on.
- Paper wastage is hurting the climate by utilizing water and trees furthermore, makes lots of trash.
- The tracking system comes at high cost.

2.3 FEASIBILITY STUDY

The feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and time that is spent on it. Feasibility study lets the developer foresee the future of the project and its usefulness. Feasibility study is a test of system proposed regarding its workability, impact on the organization, ability to meet the needs and effective use of resources. Thus when a new project is proposed, it normally goes through a feasibility study before it is approved for development.

Steps in feasibility analysis ,eight steps that involved in the feasibility analysis are: Form a project team and appoint a project leader, Prepare system flowcharts, Enumerate potential proposed system, Define and identify characteristics of proposed system, Determine and evaluate performance and cost effective of each proposed system, Weight system performance and cost data, Select the best-proposed system, Prepare and report final project directive to management.

This document provides the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as:

- Technical Feasibility
- Economic Feasibility
- Operational Feasibility
- Social Feasibility

2.3.1 Technical Feasibility

They must be evaluated from the technical viewpoint first. The assessment of this feasibility must be based on an outline design of the system requirement in terms of input, output, programs, procedures and staff. Having identified an outline system, the investigation must be go on to suggest the type of equipment, required method of developing the system, method of running the system once it has been designed.

The projects should be developed such that the necessary functions and performance are achieved within the constraints. The project is developed with the latest technology. Through the technology may become obsolete after period of time. Due to the fact that newer versions of some support older versions, the system may still be used. So that, there are only minimal constraints involved with this project.

This system has been developed using Python, which is very user friendly. Dataset is used as back end to avoid the problems in term of volume of data, trends, frequency of updating, cycles of activity etc…

2.3.2 Economical Feasibility

Developing system must be justified by cost and benefit .Criteria to ensure that effort is concentrated on project that will give best return at the earliest. One of the factors that affect the development of a new system is the cost it would require.

Because we used the system, which support Python and Dataset .Mostly, it does not cause any fault hence maintenance cost to gain. But we can make modifications based on our technological development.

2.3.3 Operational Feasibility

As estimated should be made to determine how much effort and care will go into the developing of the system including the training to be given to the user. Usually, people are reluctant to changes that that come in their progression. The computer initialization will certainly affected the multiple person live detections recognition and prediction. Hence an additional effort is to be made to train and educate the users on the new way of the system.

2.3.4 Social Feasibility

The accept of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessary. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

2.4 PROPOSED SYSTEM

The project starts by initializing the video interface through webcam. The system uses OpenCV which is open-source computer vision library to process the video. We are using cv2.VideoCapture() function to get a video capture object for the camera and will read the data from that object using VirtualAirCanvas.cap.read() for further processing. Once the video has been processed, system uses MediaPipe to detect and identify landmarks on the user's hand. Using the process () function, the system can detect the hand landmarks. We can draw or write on the canvas or screen when the "draw" tool is selected with the finger. Together with this, "Shapes Integration," or the ability to use pre-made shapes on the canvas by just selecting the desired forms/function, this is another key component of the suggested approach. We provided a variety of forms, including brush size, eraser size, color selection etc. Moreover, we can remove the artwork by simply choosing the erase tool. The user can exit the application by pressing ‘esc’ key

Advantages of Proposed System

- Fingertip tracking rather than using any object.
- MediaPipe is used for positioning of fingers.
- Inbuilt functions can be easily selected.
- Able to clear the window virtually.
- No need to touch the screen.
- No Object(cap) is required.

3. SYSTEM SPECIFICATION

3.1 HARDWARE SPECIFICATION

➢ System	:	Dual Core.
➢ Hard Disk	:	500 GB.
➢ Monitor	:	15'' LED
➢ Input Devices	:	Keyboard, Mouse
➢ Ram	:	2 GB.

3.2 SOFTWARE SPECIFICATION

➢ Development Platform	:	Windows
➢ IDE	:	Anaconda
➢ Front End	:	Tkinter, Python, Mediapipe, Opencv, TensorFlow, Numpy, Pandas
➢ Back-End	:	Dataset

3.1. FRONT END

Python

Python is a general purpose, dynamic, high level, and interpreted programming language. It supports Object Oriented programming approach to develop applications. It is simple and easy to learn and provides lots of high-level data structures.Python is easy to learn yet powerful and versatile scripting language, which makes it attractive for Application Development. Python's syntax and dynamic typing with its interpreted nature make it an ideal language for scripting and rapid application development.Python supports multiple programming pattern, including object-oriented, imperative, and functional or procedural programming styles. It is not intended to work in a particular area, such as web programming. That is why it is known as multipurpose programming language because it can be used with web, enterprise, 3D CAD, etc.We don't need to use data types to declare variable because it is dynamically typed so we can write a=10 to assign an integer value in an integer variable.

Python makes the development and debugging fast because there is no compilation step included in Python development, and edit-test-debug cycle is very fast. It is known for its general-purpose nature that makes it applicable in almost every domain of software development. Python makes its presence in every emerging field. It is the fastest-growing programming language and can develop any application.

Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. The name Pandas is derived from the word Panel Data – an Econometrics from Multidimensional data.

In 2008, developer Wes McKinney started developing pandas when in need of high performance, flexible tool for analysis of data.

Prior to Pandas, Python was majorly used for data managing and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data — load, prepare, manipulate, model, and analyze.

Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

- Fast and efficient Data Frame object with default and customized indexing.
- Tools for loading data into in-memory data objects from different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of date sets.
- Label-based slicing, indexing and subsetting of large data sets.
- Columns from a data structure can be deleted or inserted.
- Group by data for aggregation and transformations.
- High performance merging and joining of data.
- Time Series functionality.

Numpy

NumPy is a Python package. It stands for 'Numerical Python'. It is a library consisting of multidimensional array objects and a collection of routines for processing of array.

Numeric, the ancestor of NumPy, was developed by Jim Hugunin. Another package Numarray was also developed, having some additional functionalities. In 2005, Travis Oliphant created NumPy package by incorporating the features of Numarray into Numeric package. There are many contributors to this open source project.

Operations using NumPy

Using NumPy, a developer can perform the following operations −

- Mathematical and logical operations on arrays.
- Fourier transforms and routines for shape manipulation.
- Operations related to linear algebra. NumPy has in-built functions for linear algebra and random number generation.

NumPy – A Replacement for MatLab

NumPy is often used along with packages like **SciPy** (Scientific Python) and **Matplotlib** (plotting library). This combination is widely used as a replacement for MatLab, a popular platform for technical computing. However, Python alternative to MatLab is now seen as a more modern and complete programming language.

Matplot

Matplotlib is one of the most popular Python packages used for data visualization. It is a cross-platform library for making 2D plots from data in arrays.

Matplotlib is written in Python and makes use of NumPy, the numerical mathematics extension of Python.

It provides an object-oriented API that helps in embedding plots in applications using Python GUI toolkits such as PyQt, WxPythonotTkinter.

It can be used in Python and IPython shells, Jupyter notebook and web application servers also.

Matplotlib has a procedural interface named the Pylab, which is designed to resemble MATLAB, a proprietary programming language developed by MathWorks.

Matplotlib along with NumPy can be considered as the open source equivalent of MATLAB.

Matplotlib was originally written by John D. Hunter in 2003. The current stable version is 2.2.0 released in January 2018.

Matplotlib and its dependency packages are available in the form of wheel packages on the standard Python package repositories and can be installed on Windows, Linux as well as MacOS systems using the pip package manager.

Sklearn

What is Scikit-Learn (Sklearn)

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib.

Origin of Scikit-Learn

It was originally called scikits.learn and was initially developed by David Cournapeau as a Google summer of code project in 2007. Later, in 2010, Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, and Vincent Michel, from FIRCA (French Institute for Research in Computer Science and Automation), took this project at another level and made the first public release (v0.1 beta) on 1st Feb. 2010.

TKinter

Tkinter is a standard Python library for creating graphical user interfaces (GUIs). It provides a simple and efficient way to create windows, dialogs, buttons, text boxes, and other GUI elements for desktop applications. Tkinter is based on the Tk GUI toolkit, which originated as part of the Tcl scripting language. Tkinter provides a variety of widgets and layout managers for building more complex GUIs. Some commonly used widgets include buttons, entry fields, text areas, checkboxes, radio buttons, and listboxes.

Media-pipe

Media Pipe is an open-source framework developed by Google that provides a comprehensive solution for building real-time multimodal (e.g., video, audio) perceptual pipelines. It offers a wide range of pre-built components for tasks such as object detection, face detection, hand tracking, pose estimation, and more. Media Pipe aims to simplify the development of complex multimedia processing pipelines by providing reusable components that developers can integrate into their applications with ease. It supports various platforms including desktop, mobile, and embedded devices, making it suitable for a wide range of applications such as augmented reality (AR), virtual reality (VR), gesture recognition, and video analysis.

OpenCv

OpenCV is a cross-platform library using which we can develop real-time **computer vision applications**. It mainly focuses on image processing, video capture and analysis including features like face detection and object detection.

Let’s start the chapter by defining the term "Computer Vision".

Computer Vision can be defined as a discipline that explains how to reconstruct, interrupt, and understand a 3D scene from its 2D images, in terms of the properties of the structure present in the scene. It deals with modeling and replicating human vision using computer software and hardware.

Computer Vision overlaps significantly with the following fields −

- **Image Processing** − It focuses on image manipulation.
- **Pattern Recognition** − It explains various techniques to classify patterns.
- **Photogrammetry** − It is concerned with obtaining accurate measurements from images.

Image processing deals with image-to-image transformation. The input and output of image processing are both images.

Computer vision is the construction of explicit, meaningful descriptions of physical objects from their image. The output of computer vision is a description or an interpretation of structures in 3D scene.

Applications of Computer Vision

Here we have listed down some of major domains where Computer Vision is heavily used.

Robotics Application

- Localization − Determine robot location automatically
- Navigation Obstacles avoidance
- Assembly (peg-in-hole, welding, painting)

TensorFlow

TensorFlow is an open-source machine learning library developed by Google. It’s designed for a range of computational tasks, but it’s particularly known for its applications in deep neural networks. TensorFlow allows developers to create complex machine learning models with ease, thanks to its flexible architecture that supports computation across various platforms, including CPUs and GPUs. It uses data flow graphs to represent mathematical operations and the multidimensional data arrays (tensors) that flow between them, making it a powerful tool for numerical computation and large-scale machine learning.

3.2.BACK END

Dataset

The key to success in the field of machine learning or to become a great data scientist is to practice with different types of datasets. But discovering a suitable dataset for each kind of machine learning project is a difficult task. So, in this topic, we will provide the detail of the sources from where you can easily get the dataset according to your project.

Before knowing the sources of the machine learning dataset, let's discuss datasets.

What is a dataset?

A dataset is a collection of data in which data is arranged in some order. A dataset can contain any data from a series of an array to a database table. Below table shows an example of the dataset:

A tabular dataset can be understood as a database table or matrix, where each column corresponds to a **particular variable,** and each row corresponds to the **fields of the dataset.** The most supported file type for a tabular dataset is **"Comma Separated File,"** or **CSV.** But to store a "tree-like data," we can use the JSON file more efficiently.

Types of data in datasets

- **Numerical data:**Such as house price, temperature, etc.
- **Categorical data:**Such as Yes/No, True/False, Blue/green, etc.

- **Ordinal data:**These data are similar to categorical data but can be measured on the basis of comparison.

Note: A real-world dataset is of huge size, which is difficult to manage and process at the initial level. Therefore, to practice machine learning algorithms, we can use any dummy dataset.

Need of Dataset

To work with machine learning projects, we need a huge amount of data, because, without the data, one cannot train ML/AI models. Collecting and preparing the dataset is one of the most crucial parts while creating an ML/AI project.

The technology applied behind any ML projects cannot work properly if the dataset is not well prepared and pre-processed.

During the development of the ML project, the developers completely rely on the datasets. In building ML applications, datasets are divided into two parts training and testing:

Kaggle Datasets

Kaggle is one of the best sources for providing datasets for Data Scientists and Machine Learners. It allows users to find, download, and publish datasets in an easy way. It also provides the opportunity to work with other machine learning engineers and solve difficult Data Science related tasks.

Kaggle datasets encompass a wide range of topics and formats, including CSV files, JSON files, databases, images, and more. These datasets are contributed by Kaggle users, researchers, companies, and organizations, covering domains such as finance, healthcare, climate science, computer vision, natural language processing, and more. Kaggle provides tools and resources for data exploration, visualization, and analysis, making it a valuable resource for data enthusiasts, researchers, and professionals alike.

4. SYSTEM DESIGN

SYSTEM MODULES

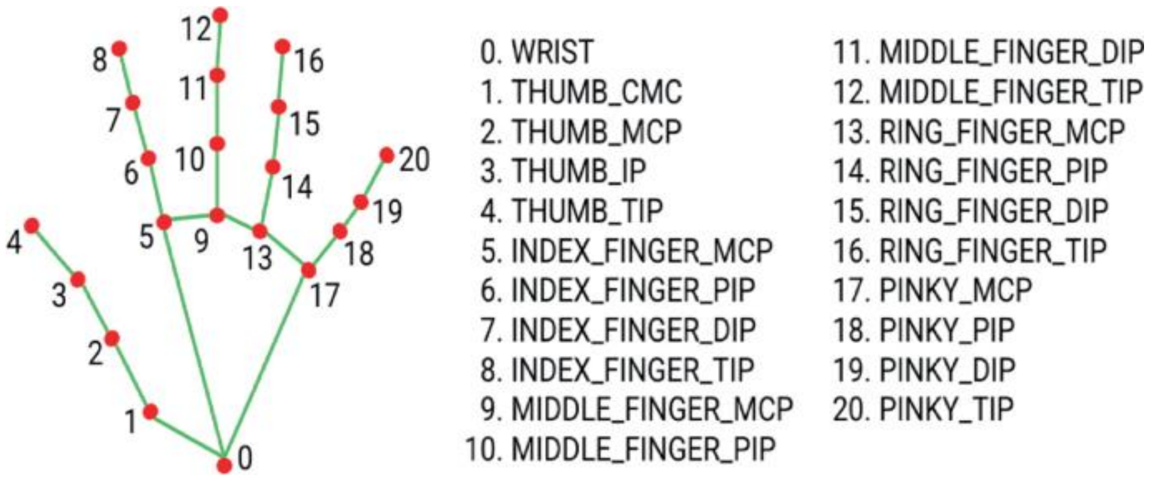
- Finger Recognition
- Writing With Free Hand
- Tools Integration
- Erase Virtually
- Air Writing Recognition

4.1 MODULE DESCRIPTION

Finger Tip Recognition

This module focuses on accurately detecting and interpreting user hand gestures and track specific points on the hand, such as fingertips, palm, and joints. Utilizes computer vision libraries such as OpenCV and MediaPipe for real-time hand tracking and gesture recognition.

It will track the movement of the finger to create a trajectory.



0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

Writing With Free Hand

This module enables users to draw or write with free hand in air in front of the webcam using natural hand gestures. Employs MEDIAPIPE for solving the issues in hand tracking. When the user points the index finger(landmark 8) on to the draw tool , the user can start drawing or writhing base on his requirement. The system will capture this motion and extracts the trajectory of air writing as sequence of coordinates or features., then displays it on the screen in real-time.

Tools Integration

This module is a key part of the Air Writing and Recognition System as it essentially enables a more interactive and intuitive user experience by allowing users to control the system's functions using simple hand gestures .It's responsible for managing the interaction between the user and the on-screen display (OSD) tools. Its functionality includes:

Tool Selection: When the user points at a specific tool on the OSD using their index and middle finger, the system recognizes this gesture and selects the corresponding tool.

Tool Activation: Once a tool is selected, the user can activate it by raising their index finger. For example, if the user selects the "draw shape" tool, they can start drawing in the air by raising their index finger.

Tool Deactivation and Switching: The tool remains active until the user selects another tool. This allows the user to perform multiple actions with the same tool without needing to reselect it each time.

Erase Virtually

Air Canvas application enables users to switch to the erasing mode by using the selection mode from the toolbar and clicking on the eraser icon. It draws a transparent line when the finger moves thus erasing colored points. By processing the video frames and selectively removing corresponding drawings from the canvas, the application creates the effect of erasing unwanted strokes.

Air Writing Recognition

This model recognizes gestures written in the air and converts it into text, users would be able to write "air words" facing a web camera either real time and have those gestures translated into letters or digits. System then locate the finger tip movement and construct an image using these coordinates and preprocess it to the format required by the neural network.

This module will use a Deep Learning model called CNN(Convolutional Neural Network) to recognize and interpret the characters written in the air by the user.

CNN algorithm is used for training the model and feature extraction through a fully connected layers that classify the features into respective classes[alphabets and digits].

Gesture Tracking: The system tracks the movement of the user’s hand as they write characters in the air. This is typically done using computer vision techniques to create a trajectory that represents the written character.

Preprocessing: The trajectory data is preprocessed to normalize it and remove any noise. This could involve rescaling the data, smoothing the trajectory, and segmenting the trajectory into individual characters.

Feature Extraction: Features are extracted from the preprocessed trajectory data. These features could include geometric properties of the trajectory, temporal properties of the writing motion, and statistical properties of the trajectory and motion.

Character Recognition: A machine learning or deep learning model is used to recognize the characters based on the extracted features. This could involve training a model on a dataset of air-written characters and then using this model to recognize characters written by the user.

Postprocessing: The recognized characters are postprocessed to correct any errors and improve the recognition accuracy

Output Generation: The recognized and postprocessed characters are then outputted to the user, completing the air writing recognition process.

4.2 INPUT DESIGN

A major part in the design of the system is the preparation of the input. Input is necessary for the successful development and implementation of the system. The input design is the link that ties in the information systems into the process of converting oriented inputs to computer based formats. The quality of the inputs determines the quality of the output. Input specifications describe the manner in which data enter the system for processing. Inaccurate input is the common cause of errors in data processing. The input design is the process of converting user oriented inputs to a computer-based format. So the input interface design is the important role in controlling errors.

This system is developed in a user-friendly manner. The forms are being designed in such a way that during the processing the cursor is placed in the position where the data must be entered. An option is to select an appropriate input from the list of values.

Validations are made for each of every data enters to a new field so that he/she understandable what is to be entered whenever the user enters an error data. Error messages are displayed.

In input design, user oriented inputs are converted into computer based format. In output design, the emphasis is on producing the hard copy of the information requested or displaying the output on a display screen in a predefined format. The following features have been incorporated into the input design of the proposed system.

Easy Data Input

Appropriate messages are provided in the message area, which prompts the user in entering the right data. Erroneous data inputs are checked at the end of each screen entry.

Data Validation

The input data is validated to minimize the errors in data entry. For certain data specific codes have been given and validation is done which enables the user to enter the required data or correct them if they entered wrong codes.

User Friendliness

User is never left in a state of confusion as to what is happening, instead appropriate error and acknowledge messages are sent. Error maps are used to indicate the error codes and specific error messages. Simple error messages are used to avoid inconvenience for users.

Consistent Format

A fixed format is adopted for displaying the title messages. Every screen has a line, which displays the operation that can be performed after the data entry. They are normally done at the touch of a key.

Interactive Dialogue

The system engages the user in an interactive dialogue. The system is able to extract missing or emitted information from the user by directing the user through appropriate messages, are displayed.

4.3 OUTPUT DESIGN

The output is the most important and direct source of information to the user. The output should be provided in most efficient formatted way. Based on the options given by the user and the administrator various types of output screens have been generated.

The computer output is the most important and the direct source of information to the user. Efficient and intelligible output design improves the system's relationship with the user and helps in decision-making. Output design was studied going actively during the study phase. The objective of the output design is defined the contents and format of all documents and reports in an attractive and useful format.

Types of Outputs

- External Outputs
- Internal outputs
- Operational outputs
- Interactive outputs

External outputs are those, whose destination will be outside the organization and which require special attention as the project image of the organization. Internal outputs are those, whose destination is within the organization. It is to be carefully designed, as they are the user's main interface with the system. Interactive outputs are those, which the user uses in communication directly with the computer.

Output generally refers to the results and information that are generated by the system. It can be in the form of operational documents and reports. The major form of output is a hard copy from the printer.

4.4 DATABASE DESIGN

A database is an organized collection of data. The data is typically organized to model relevant aspect of reality. In a way that supports processes requiring this information. A general-purpose database management system(DBMS) is a software system designed to allow the definition, creation, querying, update and administration of database. Well-known DBMS include MySQL, Microsoft access.

They actively deal with the design of the physical database. A key is to determine how to access paths are to be implemented. A physical path is derived from a logical path., pointers, chains, or other mechanism may be implemented. The general theme behind database is to handle information as a whole. The general objective is to make information access easy, quick, inexpensive and flexible for the user. Database design is the most critical part of the design phase. Database design transforms the information domain model created during the analysis into the data structures that will be required to implement the software system.

Dataset

A dataset is a collection of data in which data is arranged in some order. A dataset can contain any data from a series of an array to a database table. Below table shows an example of the dataset: A tabular dataset can be understood as a database table or matrix, where each column corresponds to a particular variable, and each row corresponds to the fields of the dataset. The most supported file type for a tabular dataset is "Comma

Separated File," or CSV. But to store a "tree-like data," we can use the JSON file more efficiently.

To work with machine learning projects, we need a huge amount of data, because, without the data, one cannot train ML/AI models. Collecting and preparing the dataset is one of the most crucial parts while creating an ML/AI project.

The technology applied behind any ML projects cannot work properly if the dataset is not well prepared and pre-processed.

During the development of the ML project, the developers completely rely on the datasets. In building ML applications, datasets are divided into two parts:

- **Training dataset:**
- **Test Dataset**

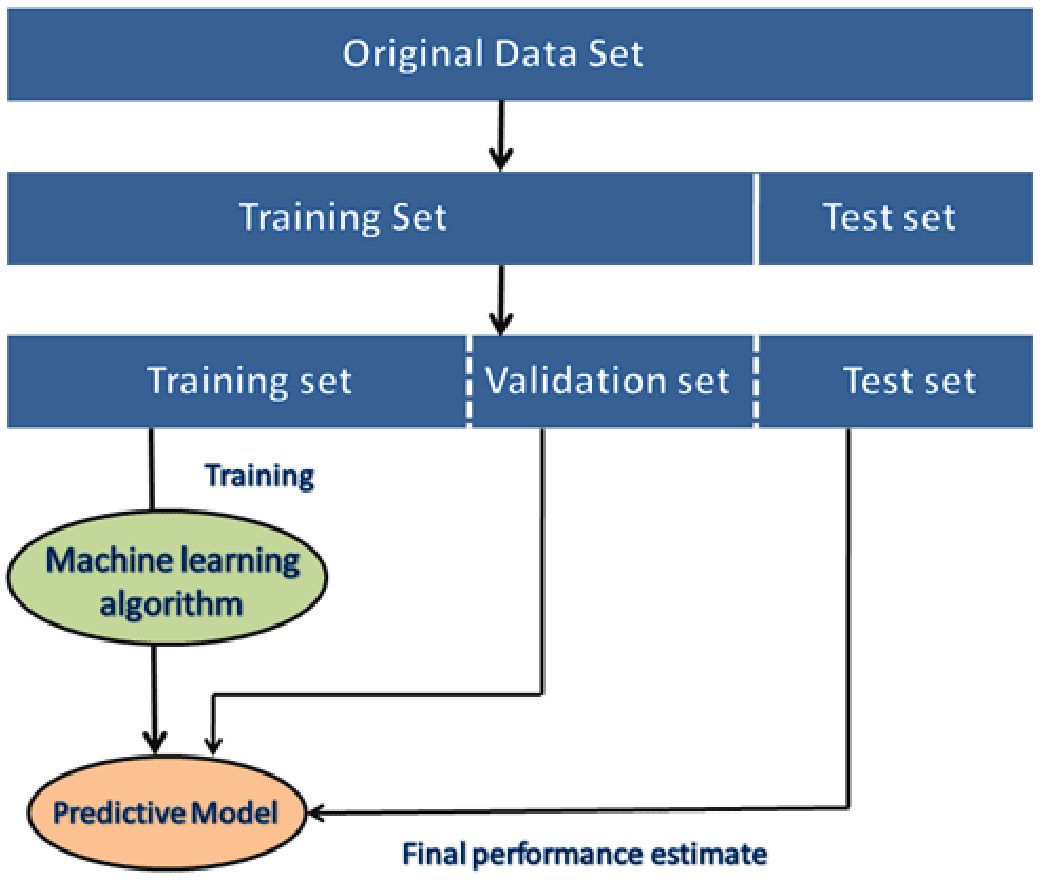


Figure 4.1 .Model Training

EMNIST Datasets

The EMNIST (Extended Modified National Institute of Standards and Technology) dataset is a set of handwritten character digits derived from the NIST Special Database 19 and converted to a 28x28 pixel image format.

EMNIST ByClass: This set contains 814,255 characters with 62 unbalanced classes1. These classes include 10 digits (0-9) and 52 alphabets (26 lowercase and 26 uppercase letters).

EMNIST ByMerge: This set also contains 814,255 characters but with 47 unbalanced classes1. It merges some classes of letters that look similar in their upper and lower case forms, such as 'C' and 'c', 'I' and 'i', 'J' and 'j', 'K' and 'k', 'L' and 'l', 'M' and 'm', 'O' and 'o', 'P' and 'p', 'S' and 's', 'U' and 'u', 'V' and 'v', 'W' and 'w', 'X' and 'x', 'Y' and 'y', 'Z' and 'z'. So, it has 10 digits (0-9) and 37 alphabets.

EMNIST Balanced: This set contains 131,600 characters with 47 balanced classes1. It has the same classes as the EMNIST ByMerge.

The Back end of AIR WRITING RECOGNITION SYSTEM was designed using EMINST dataset.It is used in following way:

Training the Model: The EMNIST dataset, which contains images of handwritten characters and digits, can be used to train a Convolutional Neural Network (CNN) or other machine learning models. The model learns to recognize the patterns of each character or digit from this dataset.

Testing the Model: A portion of the EMNIST dataset can be used to test the trained model. This helps in evaluating the model's performance and accuracy in recognizing the characters or digits.

Recognizing Air-Written Characters: Once the model is trained, it can be used to recognize characters or digits written in the air. The trajectory of the air-written character is captured, preprocessed, and then fed into the trained model for recognition.

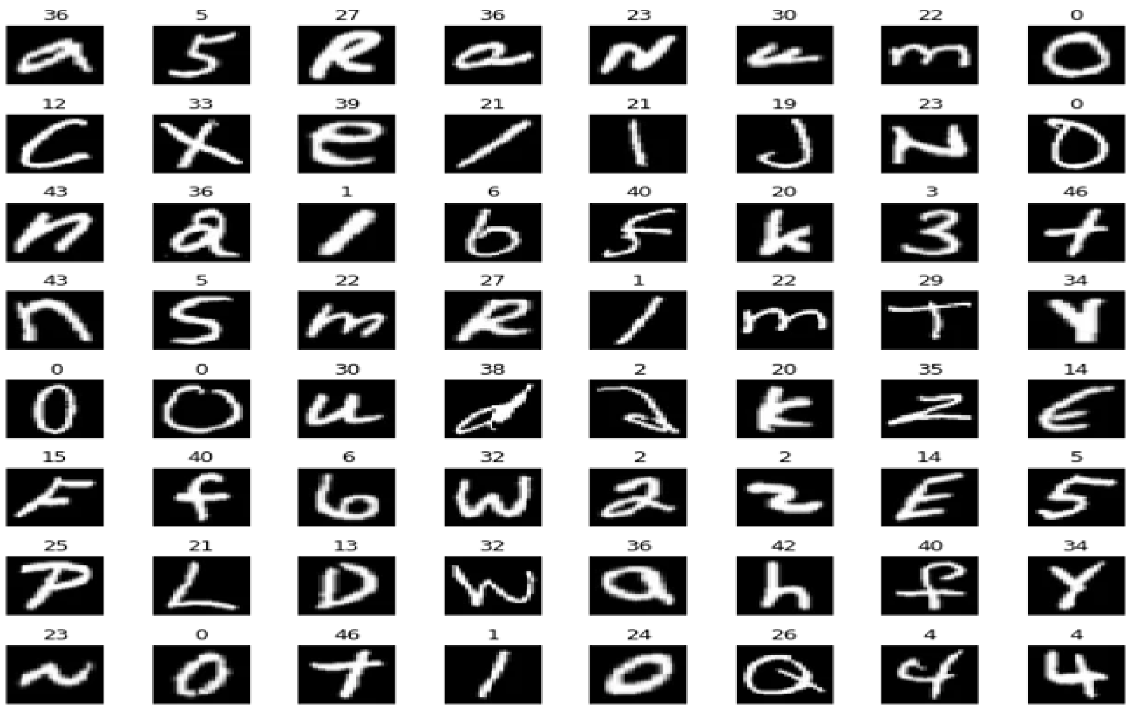


Figure 4.2 Sample of EMNIST Dataset

4.5 ARCHITECTURE DIAGRAMS

The architecture diagram is a schematic representation of a collection of ideas that are part of an architecture including its principles, elements and materials. Architecture diagram will support designers and engineers in visualizing a system or application's high-level, overarching layout to ensure the framework addresses the needs of their customers.You may also use architecture diagrams to explain patterns which are used in the design. It's a bit like a blueprint that can be used as a guide for convenience among your team to discuss, improve and follow.

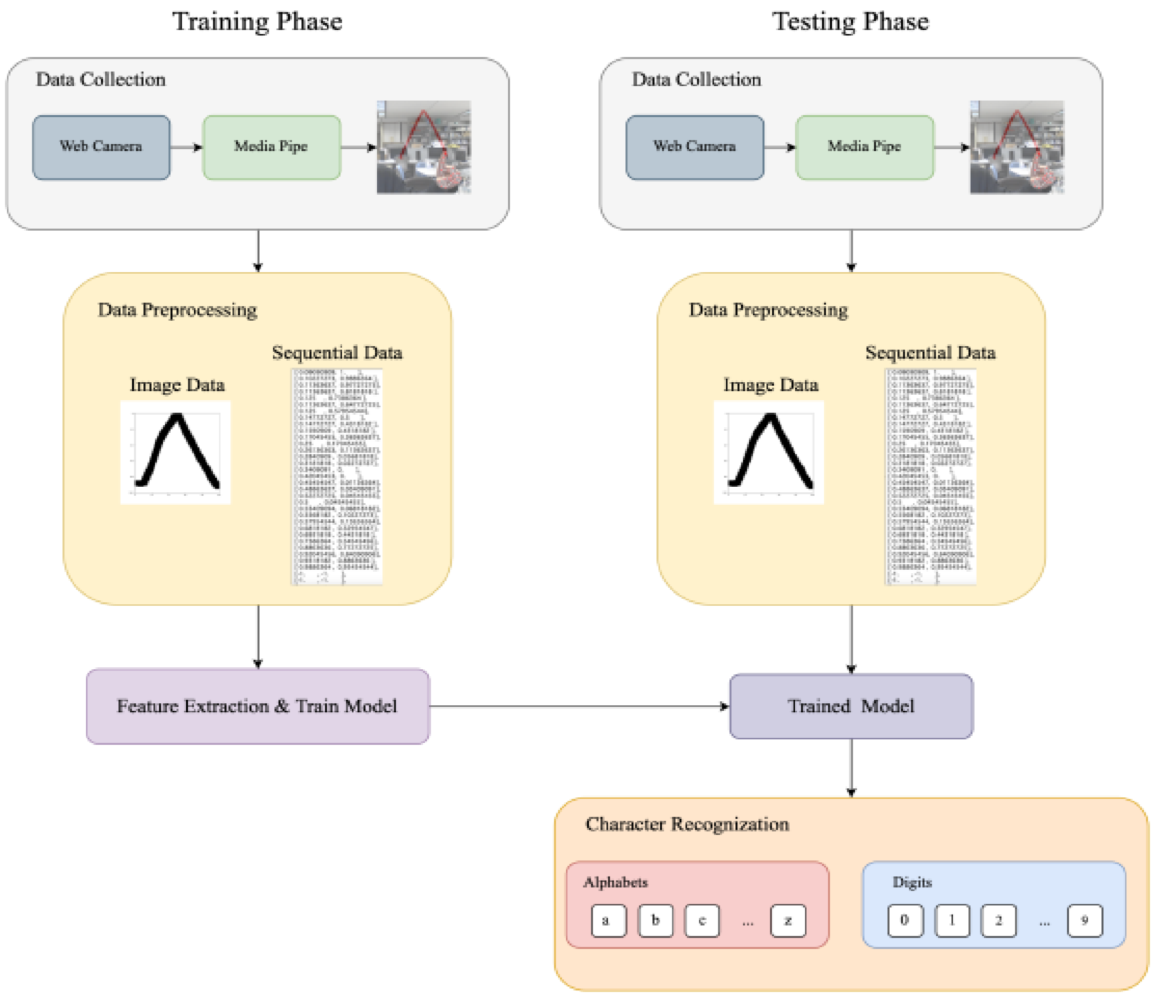


Figure 4.3. Recognition of a Letter

4.6 UML DIAGRAM

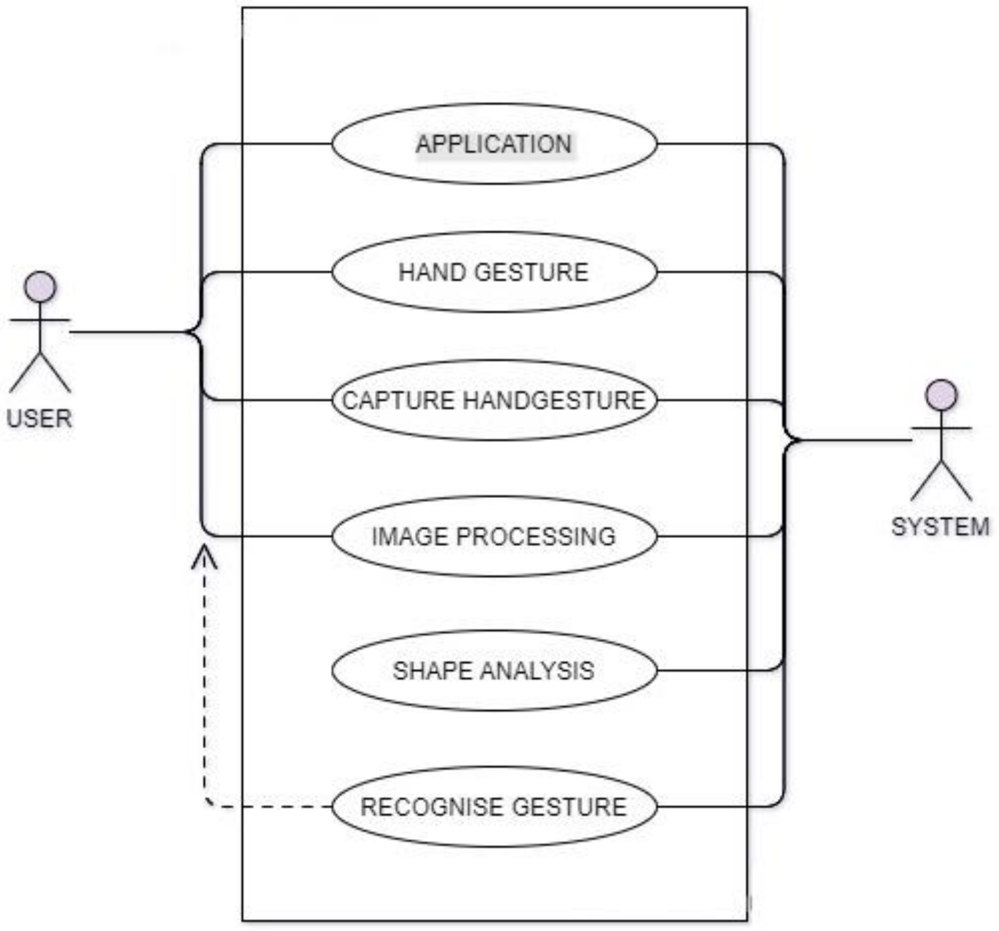
UML is a unified modeling language. Taking the SRS document of the analysis as input in the design phase of the generated UML diagrams. UML is just a language so it is only part of the software development process. The UML is process independent, although optimally it should be used in a process that should be driven, architecture-centric, iterative, and incremental. UML is the language of visualizing, extra cost, article writing in a software-centric system. This software component is based on a graphical presentation.

Folklore is a language. Vocabulary and rules focus on the ideological and material rights of the system. UML Blue coloring language is a standard language for software print. UML is a graphical language, containing all systems. The whole function of the programming space that can be represented also has different structures. These are different numbers than UML.

4.6.1 USECASE DIAGRAM

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well. The figure shows the use case diagram for the system.

The following figure shows the use case diagram:



4.6.2 SYSTEM SEQUENCE DIAGRAM

A system sequence diagram is, as the name suggests, a type of sequence diagram in UML. These charts show the details of events that are generated by actors from outside the system. Standard sequence diagrams show the progression of events over a certain amount of time, while system sequence diagrams go a step further and present sequences for specific use cases. Use case diagrams are simply another diagram type which represents a user's interaction with the system. An SSD shows – for one particular scenario of a use case –

- The events that external actors generate
- Their order
- Inter-system events

SSDs are derived from use cases; SSDs are often drawn for the main success scenarios of each use case and frequent or complex alternative scenarios. System sequence diagrams are used as input for object design.

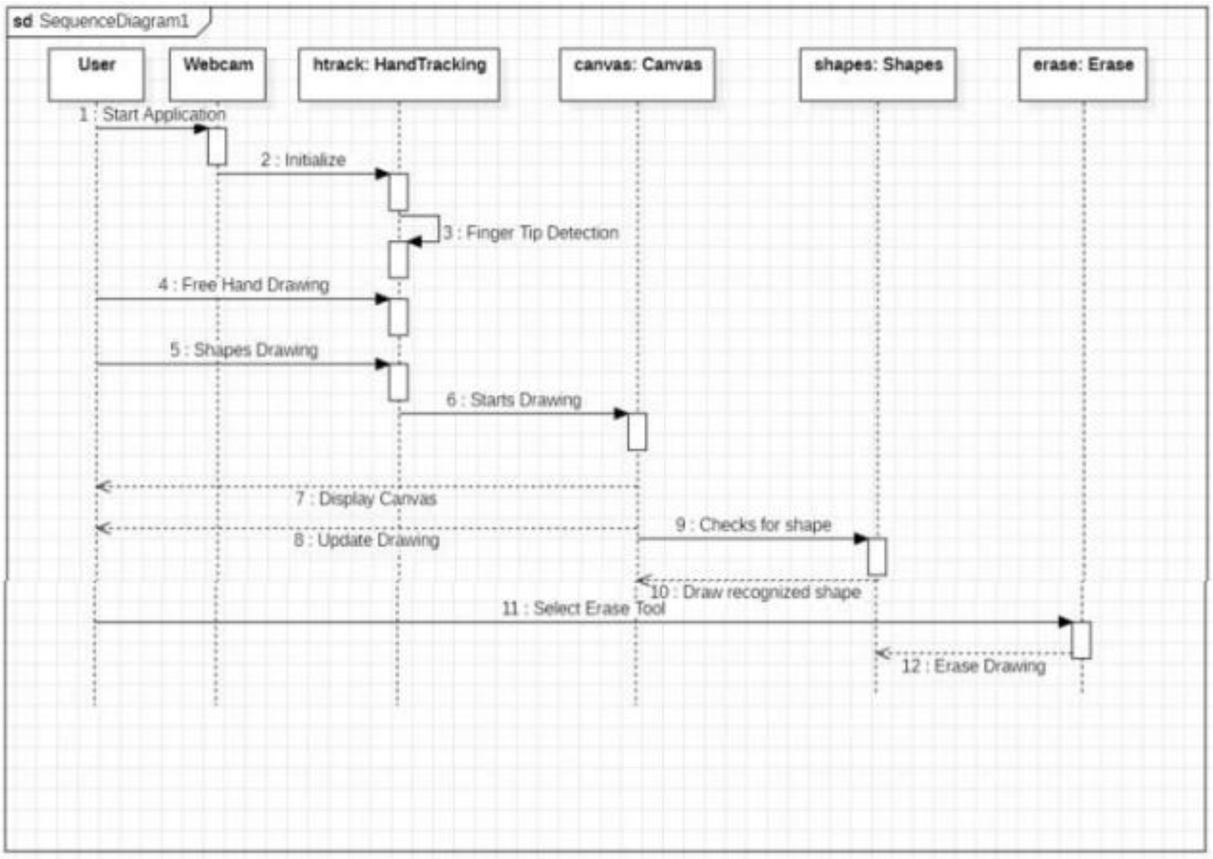


Figure 4.4 seqence diagram

4.7 DATA FLOW DIAGRAM

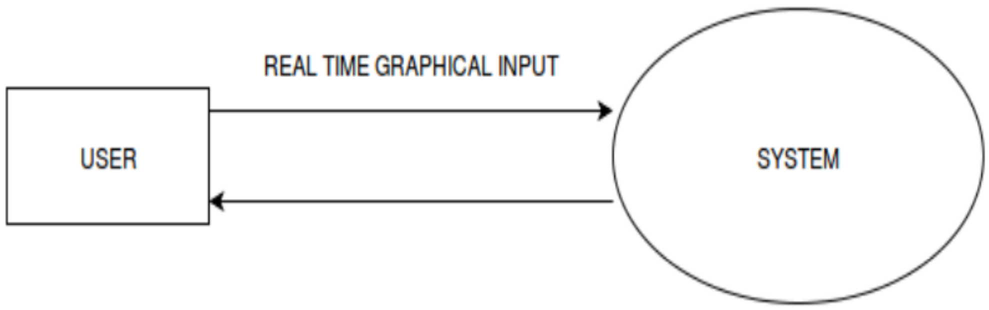
DFD is the abbreviation for **Data Flow Diagram**. The flow of data of a system or a process is represented by DFD. It also gives insight into the inputs and outputs of each entity and the process itself. DFD does not have control flow and no loops or decision rules are present. Specific operations depending on the type of data can be explained by a flowchart. It is a graphical tool, useful for communicating with users ,managers and other personnel. it is useful for analyzing existing as well as proposed system.

It should be pointed out that a DFD is not a flowchart. In drawing the DFD, the designer has to specify the major transforms in the path of the data flowing from the input to the output. DFDs can be hierarchically organized, which helps in progressively partitioning and analyzing large systems. DFDs are commonly used during problem analysis. DFDs are quite general and are not limited to problem analysis for software requirements specification.

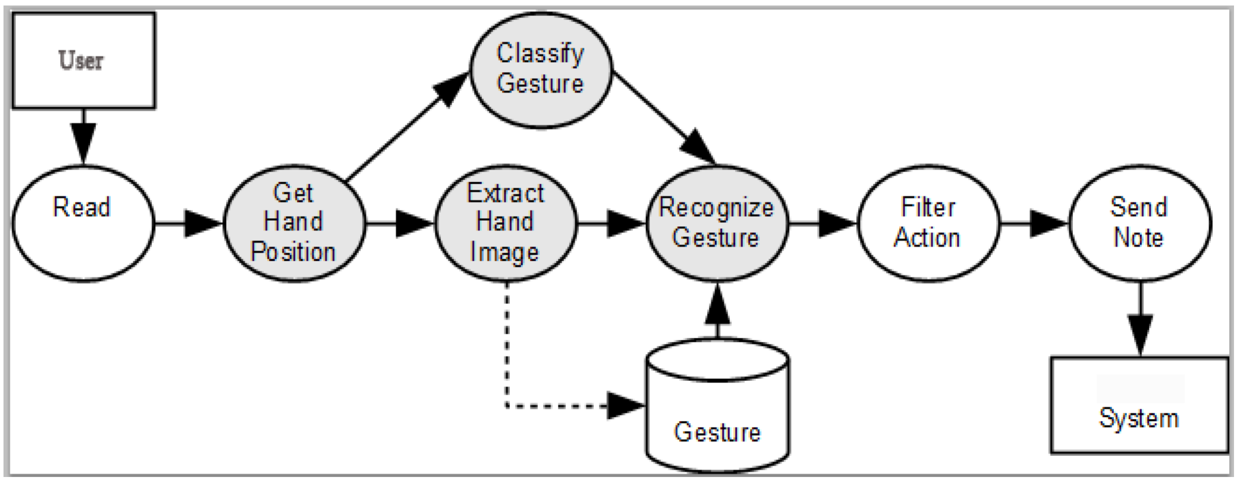
DFDs are very useful in understanding a system and can be effectively used during analysis.It views a system as a function that transforms the inputs into desired outputs. The DFD aims to capture the transformations that take place within a system to the input data so that eventually the output data is produced. The processes are shown by named circles and data flows are represented by named arrows entering or leaving the bubbles. A rectangle represents a source or sink and it is a net originator or consumer of data. A source sink is typically outside the main system of study.

The Data Flow Diagram has 4 components. Process Input to output transformation in a system takes place because of process function. The symbols of a process are rectangular with rounded corners, oval, rectangle or a circle. The process is named a short sentence, in one word or a phrase to express its essence Data Flow Data flow describes the information transferring between different parts of the systems. The arrow symbol is the symbol of data flow. A relatable name should be given to the flow to determine the information which is being moved. Data flow also represents material along with information that is being moved. Material shifts are modeled in systems that are not merely informative. A given flow should only transfer a single type of information. The direction of flow is represented by the arrow which can also be bi-directional. Warehouse The data is stored in the warehouse for later use. Two horizontal lines represent the symbol of the store. The warehouse is simply not restricted to being a data file rather it can be anything like a folder with documents, an optical disc, a filing cabinet. The data warehouse can be viewed independent of its implementation. When the data flow from the warehouse it is considered as data reading and when data flows to the warehouse it is called data entry or data updating. Terminator The Terminator is an external entity that stands outside of the system and communicates with the system. It can be, for example, organizations like banks, groups of people like customers or different departments of the same organization, which is not a part of the model system and is an external entity. Modeled systems also communicate with terminator.

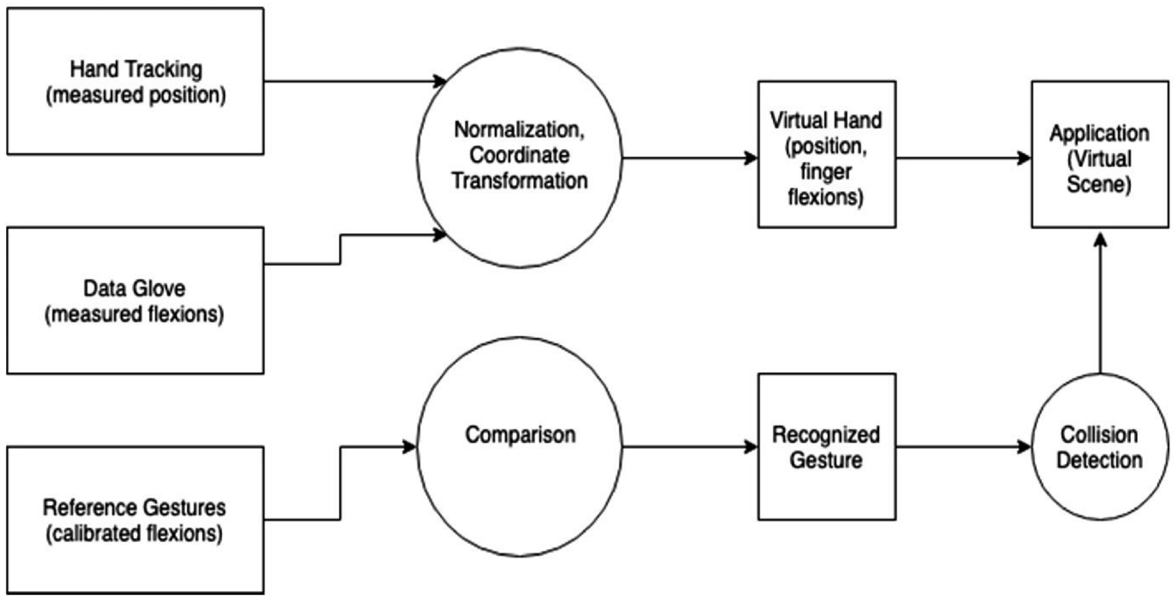
Level 0



Level 1.0

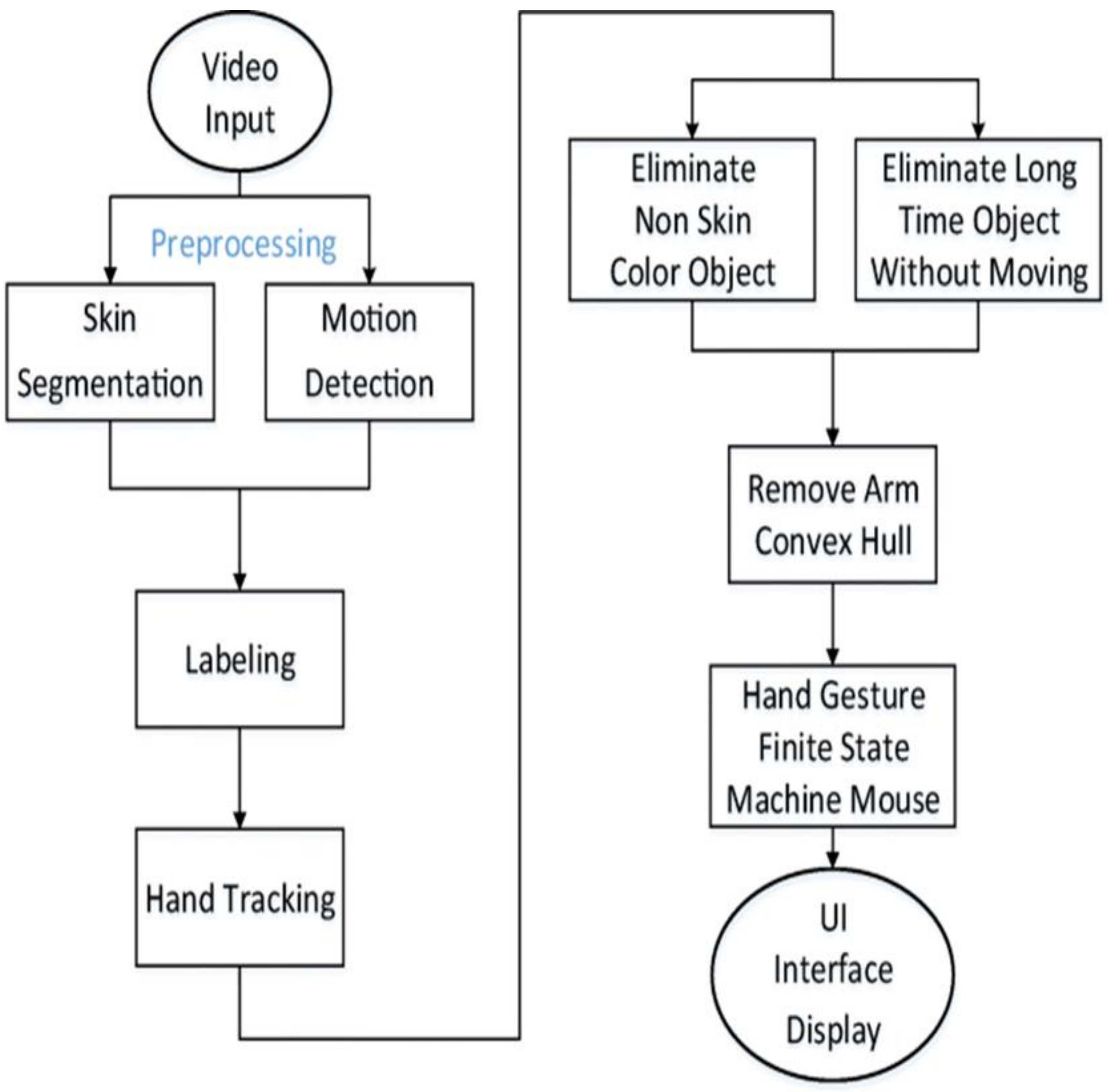


Level 1.1



Hand Tracking (measured position)
Data Glove (measured flexions)
Reference Gestures (calibrated flexions)
Normalization, Coordinate Transformation
Comparison
Virtual Hand (position, finger flexions)
Recognized Gesture
Application (Virtual Scene)
Collision Detection

Level 1.2


Video Input
Preprocessing
Skin Segmentation
Motion Detection
Labeling
Hand Tracking
Eliminate Non Skin Color Object
Eliminate Long Time Object Without Moving
Remove Arm Convex Hull
Hand Gesture Finite State Machine Mouse
UI Interface Display

5. SYSTEM DEVELOPMENT

5.1 CODING

The goal of coding or programming phase is to translate the design of the system to produce during the design phase, in to code in a programming language which can be executed by the compute, and which perform the computation specified by the design. Quality is an important goal during coding. The quality is the source code improved by the user of structured coding techniques, goo coding style and readable, consistent code format. During coding some coding standards are followed. This has two purposes, reducing the chance of making mistakes in the implementation and making it easier for someone to modify the code later on. Coding phase effects both testing and maintenance profoundly.

Coding handles each and every events or actions to be done in the software. Proper error messages are given and the validations are done in such a way the unnecessary data is not admitted during the data entry. The data, which is to be entered as such in the database, are displayed to the user in the choice wherever possible so that the user can just select from the displayed on the screen.

The coding is done in the style prescribed for the easy implementation by the company. The coding done in such a way that is reusable. This is made functions which helps in reducing the length of the code. The size of the arrays and hash tables are dynamically generated so that unnecessary space allocation is not wasted.

6. SYSTEM TESTING

Software testing is an activity to check whether the actual results match the expected results and to ensure that the software system is Defect free. It involves execution of a software component or system component to evaluate one or more properties of interest.

Software testing also helps to identify errors, gaps or missing requirements in contrary to the actual requirements. It can be either done manually or using automated tools. Some prefer saying Software testing as a white box and Black Box Testing.

6.1 UNIT TESTING

The first level of testing is called as unit testing. Here the different modules are tested and the specification produced during design for the modules. Unit testing is essential for verification of the goal and to test the internal logic of the modules. Unit testing is conducted to different modules of the project. Errors were noted down and corrected down immediately and the program clarity was increased. The testing was carried out during the programming stage itself. In this step each module is found to be working satisfactory as regard to be expected out from the module.

6.2 INTEGRATION TESTING

The second level of testing includes integration testing. It is a systematic testing of constructing structure. At the same time tests are conducted to uncover errors with the interface. It need not to be the case, that software whose modules when run individually showing results will also show perfect results when run as a whole.

The individual modules are tested again and the results are verified. The goal is to see if the modules integrated between the modules. This testing activity can be considered as testing the design and emphasizes on testing modules interaction.

6.3 VALIDATION TESTING

The next level of testing is validation testing. Here the entire software is tested. The reference document for this process is the requirement and the goal are to see if the software meets its requirements.

The requirement document reflects and determines whether the software functions as the user expected. At culmination of integration testing, software is completely assembled as a package and corrected and a final series of software test validation test begins. The proposed system under construction has been tested by using validation testing and found to be working satisfactory.

Data validation checking is done to see whether the corresponding entries made in different tables are done correctly. Proper validation checks are done in case of insertion and updating of tables, in order to see that no duplication of data has occurred. If any such case arises proper warning message will be displayed. Double configuration is done before the administrator deletes a data in order to get positive results and to see that data have been deleted by accident.

6.4 FUNCTIONAL TESTING

Functional testing is basically defined as a type of testing that verifies that each function of the software application works in conformance with the requirement and specification. This testing is not concerned with the source code of the application. Each functionality of the software application is tested by providing appropriate test input, expecting the output, and comparing the actual output with the expected output. This testing focuses on checking the user interface, APIs, database, security, client or server application, and functionality of the Application Under Test. Functional testing can be manual or automated.

- Ester does verification of the requirement specification in the software application.
- After analysis, the requirement specification tester will make a plan.
- After planning the tests, the tester will design the test case.
- After designing the test, case tester will make a document of the traceability matrix.
- The tester will execute the test case design.
- Analysis of the coverage to examine the covered testing area of the application.
- Defect management should do to manage defect resolving.

6.5 SYSTEM TESTING

System testing is a stage of implementation that aims the assurance that the system works accurately and efficiently before live operation commences. System testing makes logical assumption that all the parts of the system are correct; the goal will be successfully achieved. The testing phase is an important part of software development. It performs a critical role for quality assurance and for ensuring the reliability of the software. It is the process of finding errors and missing operations and also a complete verification to determine whether the objectives are met the user requirements are satisfied. The goal of testing is to cover requirements, design or coding errors programs. Consequently, different levels of testing are employed.

The project “Air Writing In Future Writer Using Deep Learning And AI” has done the system testing. All modules are combined together to form the entire system and the errors occurred are cleared subsequently.

6.6 WHITE BOX TESTING

White box testing is known as Clear Box testing, code-based testing, structural testing, extensive testing, and glass box testing, transparent box testing. It is a software testing method in which the internal structure/design/ implementation tested known to the tester. The white box testing needs the analysis of the internal structure of the component or system.

6.7 BLACK BOX TESTING

It is also known as behavioral testing. In this testing, the internal structure/ design/ implementation not known to the tester. This type of testing is functional testing. Why we called this type of testing is black-box testing, in this testing tester, can't see the internal code. For example, A tester without the knowledge of the internal structures of a website tests the web pages by using the web browser providing input and verifying the output against the expected outcome.

6.8 USABILITY TESTING

Unit testing is a software testing method that focuses on individual units or components of a software system. Unit testing for an Air Writing and Recognition System would involve validating individual components of the system to ensure they function as expected.

For instance, the fingertip detection could be tested by providing a variety of images and videos with different lighting conditions, hand positions, and backgrounds to ensure the system accurately identifies the fingertip. The media pipe module could be tested by simulating real-time video input and checking if it correctly detects the fingertip in each frame.

This way, unit testing helps ensure that each component of the Air Writing and Recognition System works correctly in isolation.

7. SYSTEM IMPLEMENTATION

7.1 IMPLEMENTATION

Implementation is the phase which involves the process of converting a new system design into an operational one it is the key stage in achieving a successful new system implementation includes all those activities that take place to convert from the old system to the new. The main aim of this implementation plan is to see whether the systems created and inputs were given according to the user requirements and specification that have been established a proper is essential to provide a reliable system to meet the requirements of the organization. An improper installation may affect the success of the computerized system.

Implementation is the stage of the project, where the theoretical term turned into a working system. It involves careful planning, investigation of the current system and its constraints on implementation, design of method to achieve the change over procedures and evaluation of change over methods.

The implementation plan consists of the following steps:

- Testing the developed system with sample.
- Detection and correction of errors
- Making necessary changes in the system
- Training and involvement of user personal
- Installation of hardware and software utilities

Implementation Procedure

The implementation phase is less creative than system design. A system project may be dropped at any time prior to implementation, although it becomes more difficult when it goes to the design phase. The final report to the implementation phase includes procedural, record lowest and a workable plan for implementing the candidate system.

Implementation is used to the process of converting a new or reversed system design in to and operational one. Conversion is one aspect of implementation phase. There are several methods for handling the implementation and the consequent converse from old to the new computerized system. The most secure method for conversion from the old system to the new system is to run the old and new system in parallel. In this approach, a person may operate in the manual older processing system as well as start operating the new computerized system. This method offers high security because even if there is a flow in the computerized system we can depend upon the manual system.

Implementation plan

The implementation plan includes a description of all the activities that must occur to implement the new system and to put it in to operation. It identifies the personal responsible for the activities and prepares a time chart for implementing the system.

The implementation plan consists the following steps:

- List all files required for implementation.
- Identify all data required to build new files during the implementation.
- List all documents and procedures that go in to the new system

The implementation plan should anticipate possible problem and must be able to deal with them. The usual problem may be missing documents, mixed data formats between current files, errors in data translation, missing data etc.

There are different types of implementations used in Project management, of which we use Parallel run concept. In parallel run implementation, we operate both the old and new versions of systems and processes until there is confidence that the new version is ready to support user objectives.

Python

Python is a general purpose, dynamic, high level, and interpreted programming language. It supports Object Oriented programming approach to develop applications.It is simple and easy to learn and provides lots of high-level data structures.Python is easy to learn yet powerful and versatile scripting language, which makes it attractive for Application Development.Python's syntax and dynamic typing with its interpreted nature make it an ideal language for scripting and rapid application development.Python supports multiple programming pattern, including object-oriented, imperative, and functional or procedural programming styles.

Python is not intended to work in a particular area, such as web programming. That is why it is known as multipurpose programming language because it can be used with web, enterprise, 3D CAD, etc.We don't need to use data types to declare variable because it is dynamically typed so we can write a=10 to assign an integer value in an integer variable.Python makes the development and debugging fast because there is no compilation step included in Python development, and edit-test-debug cycle is very fast.Python is known for its general-purpose nature that makes it applicable in almost every domain of software development. Python makes its presence in every emerging field. It is the fastest-growing programming language and can develop any application.

Machine Learning

Machine learning (ML) is a field devoted to understanding and building methods that let machines "learn" – that is, methods that leverage data to improve computer performance on some set of tasks. Machine learning algorithms build a model based on sample data, known as training data, in order to make predictions or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications, such as in medicine, email filtering, speech recognition, agriculture, and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks.

Supervised Learning

Supervised learning is a type of machine learning method in which we provide sample labeled data to the machine learning system in order to train it, and on that basis, it predicts the output. The system creates a model using labeled data to understand the datasets and learn about each data, once the training and processing are done then we test the model by providing a sample data to check whether it is predicting the exact output or not. The goal of supervised learning is to map input data with the output data. The supervised learning is based on supervision, and it is the same as when a student learns things in the supervision of the teacher.

Handwriting analysis could be a classic, wellexplored downside within the realm of introductory Machine Learning that encompasses several necessary topics of neural networks. Air Writing is particularly helpful for user inter phases that don't permit the user to kind on a keyboard or pen a trackpad/ touch screen, or for test input for sensible system management, among several applications. Air Writing differs from typical handwriting; the one that performs Air Writing will solely use associate degree unreal coo-ordinate to guide the writing motion. The variability of motion knowledge represents a letter is so significantly broader in Air Writing than in paper writing. Our project aims to use a mixture of computer vision and hand writing recognition to form a system that acts as a virtual white board. model users would be ready to write “air words” facing a web camera either real-time or in advance and have those gestures translated into letters or digits. The aim of the project is to further explore the task of classifying hand written text and to convert hand written texts into a digital format and achieving a virtual white board system at a cost that is accessible to the average user. We want to introduce alternative interfaces for communication that have high affordability, usability, and accessibility.

Writing in Air is a Fascinating thing. This can be achieved by Computer Vision Techniques . Air Canvas is a novel way of painting virtually without using a physical brush or canvas. It is made of system that can catch movements of artists and can draw even without touching keyboard, mouse, or touchpad. Here fingertip detection and tracking using MediaPipe are used to achieve the objective. The colour marker is detected on index finger and Object tracking is considered an important task within the field of Computer Vision. Hand Tracking and Shapes Drawing is done using Computer Vision frameworks in python.

Deep Learning

Deep learning is a sophisticated branch of machine learning that employs artificial neural networks with multiple layers to discover intricate patterns and relationships within data. These networks are inspired by the human brain and are capable of learning from vast amounts of data without explicit programming. Deep learning has revolutionized fields like image recognition, natural language processing, and autonomous driving by automatically identifying features and making predictions based on learned representations of data.

Deep learning is integral to the “Air Writing and Recognition System” project, enabling the system to accurately recognize and translate air-written characters into text.

7.2 MAINTENANCE

The term “Software Maintenance” is used to describe the software engineering activities that occur following delivery of a software product to the customer. The maintenance phase of a software life cycle is the time period in which a product performs usual work. Maintenance is classified into four types

- Corrective maintenance
- Adaptive maintenance
- Perfective maintenance
- Preventive maintenance

The software package is delivered into the customer the following problem occurs.

- The user discovers new errors in the software
- The customer wants to upgrade his hardware, thus necessitating software changes.
- New needs of user request for the functional enhancement of software.

Corrective maintenance

Corrective maintenance means repairing processing or performance failures or making changes because of previously uncorrected problems or false assumptions. Here the application is tested and reviewed for the performance and the necessary changes are made. In general modular program with structured programming is easy to correct and maintain.

Adaptive maintenance

Adaptive maintenance means changing the program function. These may involve moving the software to different machine or for instance, modifying the software to accommodate a new telecommunication protocol or additional disk drivers.

The environmental changes include:

- Hardware changes
- Operating system changes

Perfective maintenance

Perfective maintenance means enhancing the performance or modifying the programs to respond to the users additional or changing needs. Like adaptive maintenance it may require fundamental changes of the system. Perfective maintenance accounts for majority of all efforts expected on software.

Preventive maintenance

Preventive maintenance is the only maintenance activity. This is carried out without formal maintenance request from the user. When a software company realizes that the methodologies used in program have become obsolete, it may decide to develop or modify parts of the program, which don’t conform to the current trends.

Of these types, more time and money is spending on perfective than on corrective and adaptive maintenance together.

8. CONCLUSION

In conclusion, the “Air Writing and Recognition System” project opens up new possibilities for human-computer interaction, providing a novel, intuitive, and efficient method for users to interact with digital systems. The “Air Writing and Recognition System” project represents a significant advancement in the field of artificial intelligence and human-computer interaction. By combining Computer Vision, Deep learning, and natural language processing techniques, the system enables users to write in the air and convert their gestures into digital text1.The framework can possibly challenge traditional writing methods . It denies the need to convey a cell phone in Hand to record notes , giving the straightforward method for doing the same . It will also fill an extraordinary need in aiding particularly abled individuals communicate effectively . Indeed, even Senior resident or individuals who find it hard to utilize consoles will actually want to utilize the framework easily . Expanding magnificent usefulness , framework can likewise be utilized to control IOT gadgets shortly . Attracting the air can likewise be made conceivable .

The framework will be a fantastic programming for smart wearables utilizing which individuals could more readily interface with the Digital world . Augmented Reality can make text wake up. There are some constraints of the framework which can be worked on from now on. First and foremost, utilizing a handwriting recognizer instead of a character recognizer will permit the client to compose quicker. Also , our framework in some cases perceives fingertips behind the scenes and alters their state. Air-composing frameworks ought to just obey their expert's control motions and ought not be deceived by individuals around. Later on, progresses in Artificial Intelligence will improve the productivity of air writing.

<u>9. FURTHER SCOPE OF THE PROJECT</u>

It will produce an effective communication between peoples that will reduce the uses of laptops and mobile phones by abolishing the writing need. The major scope is in the teaching field while teaching online or teaching on screen. Without the mouse or any markers, we can easily implement on the screen. It will use in the designing purposes to create the immersive or interactive designs.

Gesture & Shapes Vocabulary: Enhance the system by adding more gestures and shapes to increase its functionality. Recognize a broader range of gestures for increased user convenience. Multimodal Integration: Explore the integration of voice commands to create a multi modal HCI system. This allows users to combine hand gestures with voice instructions for a richer interaction experience. Cross-Platform Compatibility: Extend compatibility to various operating systems and applications, making the system adaptable to different user needs.

User Customization: Develop features that enable users to customize gesture definitions, making the system more personalized and versatile.

10. BIBLIOGRAPHY

[1] S.V. Aswin Kumer⇑ , P. Kanakaraja, Sheik Areez, Yamini Patnaik, Pamarthi Tarun Kumar, " An implementation of virtual white board using open CV for virtual classes.

[2] Mishra, P., & Uniyal, A. (2021). Virtual Ink Using Python (No. 5707). Easy Chair.

[3] Saoji , Saurabh & Dua, & Vidyapeeth, Bharati & Choudhary, Akash & Phogat, Bharat. (2021). AIR CANVAS APPLICATION USING OPENCV AND NUMPY IN PYTHON. International Journal of Research in Engineering and Technology. 8. 2395-0056.

[4] Srungavarapu, Pranavi & Maganti, Eswar & Sakhamuri, Srilekkha & Veerada, Sai & Chinta, Anuradha. (2021). Virtual Sketch using Open CV. International Journal of Innovative Technology and Exploring Engineering. 10. 107-108. 10.35940/ijitee.H9262.0610821.

[5] Kaur, Harneet & Reddy, Busireddy & Sai, Guna & Raj, Akula. (2021). A Comprehensive overview of AR/VR by Writing in Air. International Journal of Scientific Research in Computer Science, Engineering and Information Technology. 477-482. 10.32628/CSEIT217294.

[6] M. S. Alam, K. -C. Kwon and N. Kim, "Trajectory-Based Air-Writing Character Recognition Using Convolutional Neural Network," 2019 4th International Conference on Control, Robotics and Cybernetics (CRC), 2019, pp. 86-90, doi: 10.1109/CRC.2019.00026.

[7] Yuan-Hsiang Chang , Chen-Ming Chang , "Automatic Hand Pose Trajectory Tracking System Using Video Sequences" , INTECH , pp. 132-152 , Croatia , 2019 .

[8] J. Patel, U. Mehta, K. Panchal, D. Tailor and D. Zanzmera, "Text Recognition by Air Drawing," 2021 Fourth International Conference on Computational Intelligence and Communication Technologies (CCICT), 2021, pp. 292-295, doi: 10.1109/CCICT53244.2021.00061.

[9] Z. -T. Liu, D. P. Y. Wong and P. H. Chou, "An Imu-Base Wearable Ring for On-Surface Handwriting Recognition," 2020 International Symposium on VLSI Design,

Automation and Test (VLSI-DAT), 2020, pp. 1-4, doi: 10.1109/VLSIDAT49148.2020.9196479.

[10] S. Tanaka, M. Takuma, Y. Tsukada, Recognition of Proc. 76th Nat. Conv. (IPSJ), 2014, vol. 76. no. 2, pp. 2275– 2276.

[11] T. Miyake, D. Wakatuki, I. Naito, ‘‘A basic study on recognizing fingerspelling with hand movements by the use of depth image,’’ (in Japanese), Nat. Univ. Corporation Tsukuba, Univ. Technology, Tsukuba, Japan, Tech. Rep., Dec. 2012, vol. 20, no. 1, pp. 7–13.

[12] D. Takabayashi et al., Training system for learning finger alphabets with feedback functions (in Japanese), IEICE (HIP), Tech. Rep. 112 (483) (2013) 79– 84.

[13] M. Maehatake, M. Nishida, Y. Horiuchi, A. Ichikawa, A study on sign language recognition based on gesture components of position and movement, (in Japanese), in Proc. Workshop Interact. Syst. Softw. (WISS), Japan, Dec. 2007, pp. 129–130.

[14] A. Sato, K. Shinoda, S. Furui, ‘Sign language recognition using time-of-flight camera (in Japanese), Proc. Meeting Image Recognition. Understand. (MIRU) 3 (44) (2010) 1861–1868.

[16] Y. Nishimura, D. Imamura, Y. Horiuchi, K. Kawamoto, T. Shinozaki, S. Kuroiwa, HMM sign language recognition using kinect and particle filter, (in Japanese), IEICE (PRMU), Tech. Rep. 111 (430) (2012) 161–166.

[15] M. Ohkura, R. Manabe, H. Shimada, Y. Shimada, A recognition algorithm of numerals written in the air by a finger-tip (in Japanese), J. IPSJ 52 (2) (2011) 910–916.

11. APPENDIX

11.1 SOURCE CODE

Project.py

```
# -*- coding: utf-8 -*-

import tkinter as tk
from tkinter import Message ,Text
from tkinter import *
import tkinter.messagebox
import PIL
from PIL import ImageTk
from PIL import Image
import os
import subprocess
import webbrowser

window = tk.Tk()
window.title("")

dialog_title = 'QUIT'
dialog_text = 'Are you sure?'
```

```
#window.geometry('1280x720')
#window.configure(background='#3b5999')

bg= ImageTk.PhotoImage(file="./bg.jpg")
#Create a canvas
canvas= Canvas(window,width= 400, height= 200)
canvas.pack(expand=True, fill= BOTH)
#Add the image in the canvas
canvas.create_image(0,0,image=bg, anchor="nw")

window.wm_attributes("-transparentcolor", 'grey')

window.attributes('-fullscreen', True)

window.grid_rowconfigure(0, weight=1)
window.grid_columnconfigure(0, weight=1)



def awrite():
        os.system("python canvay.py")
        exit()

def awriterec():
        os.system("python vwrtite.py")
        exit()

appwr = tk.Button(window, text="Write", command=awrite  ,fg="white"  ,bg="#6699cc"
,width=20  ,height=2, activebackground = "#99ccff" ,font=('times', 15, ' bold '))
```

```
appwr.place(x=1100, y=370)

appwrr = tk.Button(window, text="Write And Rec", command=awriterec  ,fg="white" ,bg="#6699cc"  ,width=20  ,height=2, activebackground = "#99ccff" ,font=('times', 15, ' bold '))
appwrr.place(x=1100, y=520)

appquit = tk.Button(window, text="Quit", command=window.destroy   ,fg="white" ,bg="#6699cc"  ,width=20  ,height=2, activebackground = "#99ccff" ,font=('times', 15, ' bold '))
appquit.place(x=1100, y=670)



window.mainloop()
```

canvayhand.py

```
import mediapipe as mp
import numpy as np
import cv2

class HandTracker():
    def __init__(self, mode=False, maxHands=2, detectionCon=0.5, trackCon=0.5):
        self.mode = mode
        self.maxHands = maxHands
        self.detectionCon = detectionCon
        self.trackCon = trackCon

        self.mpHands = mp.solutions.hands
        self.hands = self.mpHands.Hands(self.mode, self.maxHands, self.detectionCon,
self.trackCon)
        self.mpDraw = mp.solutions.drawing_utils

    def findHands(self, img, draw=True):
        imgRGB = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        self.results = self.hands.process(imgRGB)

        if self.results.multi_hand_landmarks:
            for handLm in self.results.multi_hand_landmarks:
                if draw:
                    self.mpDraw.draw_landmarks(img,                    handLm,
self.mpHands.HAND_CONNECTIONS)
        return img

    def getPostion(self, img, handNo = 0, draw=True):
        lmList =[]
        if self.results.multi_hand_landmarks:
            myHand = self.results.multi_hand_landmarks[handNo]
            for lm in myHand.landmark:
                h, w, c = img.shape
                cx, cy = int(lm.x*w), int(lm.y*h)
                lmList.append((cx, cy))
                if draw:
                    cv2.circle(img, (cx, cy), 5, (255,0,255), cv2.FILLED)
        return lmList

    def getUpFingers(self, img):
        pos = self.getPostion(img, draw=False)
        self.upfingers = []
        if pos:
            #thumb
            self.upfingers.append((pos[4][1] < pos[3][1] and (pos[5][0]-pos[4][0]> 10)))
            #index
            self.upfingers.append((pos[8][1] < pos[7][1] and pos[7][1] < pos[6][1]))
            #middle
            self.upfingers.append((pos[12][1] < pos[11][1] and pos[11][1] < pos[10][1]))
            #ring
            self.upfingers.append((pos[16][1] < pos[15][1] and pos[15][1] < pos[14][1]))
            #pinky
            self.upfingers.append((pos[20][1] < pos[19][1] and pos[19][1] < pos[18][1]))
        return self.upfingers
```

Canvas.py

```
from canvayhand import *
import cv2
import mediapipe as mp
import numpy as np
import random


class ColorRect():
    def __init__(self, x, y, w, h, color, text='', alpha = 0.5):
        self.x = x
        self.y = y
        self.w = w
        self.h = h
        self.color = color
        self.text=text
        self.alpha = alpha



    def drawRect(self, img, text_color=(255,255,255),
fontFace=cv2.FONT_HERSHEY_SIMPLEX, fontScale=0.8, thickness=2):
        #draw the box
        alpha = self.alpha
        bg_rec = img[self.y : self.y + self.h, self.x : self.x + self.w]
        white_rect = np.ones(bg_rec.shape, dtype=np.uint8)
        white_rect[:] = self.color
        res = cv2.addWeighted(bg_rec, alpha, white_rect, 1-alpha, 1.0)

        # Putting the image back to its position
        img[self.y : self.y + self.h, self.x : self.x + self.w] = res

        #put the letter
        tetx_size = cv2.getTextSize(self.text, fontFace, fontScale, thickness)
        text_pos = (int(self.x + self.w/2 - tetx_size[0][0]/2), int(self.y + self.h/2 + 
tetx_size[0][1]/2))
        cv2.putText(img, self.text,text_pos , fontFace, fontScale,text_color, thickness)



    def isOver(self,x,y):
        if (self.x + self.w > x > self.x) and (self.y + self.h> y >self.y):
            return True
        return False



#initilize the habe detector
detector = HandTracker(detectionCon=1)

#initilize the camera
cap = cv2.VideoCapture(0)
cap.set(3, 1280)
cap.set(4, 720)

# creating canvas to draw on it
canvas = np.zeros((720,1280,3), np.uint8)

# define a previous point to be used with drawing a line
px,py = 0,0
#initial brush color
color = (255,0,0)
#####
brushSize = 5
```

```
eraserSize = 20
####

########### creating colors ########
# Colors button
colorsBtn = ColorRect(200, 0, 100, 100, (120,255,0), 'Colors')

colors = []
#random color
b = int(random.random()*255)-1
g = int(random.random()*255)
r = int(random.random()*255)
print(b,g,r)
colors.append(ColorRect(300,0,100,100, (b,g,r)))
#red
colors.append(ColorRect(400,0,100,100, (0,0,255)))
#blue
colors.append(ColorRect(500,0,100,100, (255,0,0)))
#green
colors.append(ColorRect(600,0,100,100, (0,255,0)))
#yellow
colors.append(ColorRect(700,0,100,100, (0,255,255)))
#erase (black)
colors.append(ColorRect(800,0,100,100, (0,0,0), "Eraser"))

#clear
clear = ColorRect(900,0,100,100, (100,100,100), "Clear")
```

```
########## pen sizes #######
pens = []
for i, penSize in enumerate(range(5,25,5)):
    pens.append(ColorRect(1100,50+100*i,100,100, (50,50,50), str(penSize)))

penBtn = ColorRect(1100, 0, 100, 50, color, 'Pen')

# white board button
boardBtn = ColorRect(50, 0, 100, 100, (255,255,0), 'Board')

#define a white board to draw on
whiteBoard = ColorRect(50, 120, 1020, 580, (255,255,255),alpha = 0.6)

coolingCounter = 20
hideBoard = True
hideColors = True
hidePenSizes = True

while True:

    if coolingCounter:
        coolingCounter -=1
        #print(coolingCounter)

    ret, frame = cap.read()
    if not ret:
        break
    frame = cv2.resize(frame, (1280, 720))
    frame = cv2.flip(frame, 1)

    detector.findHands(frame)
    positions = detector.getPostion(frame, draw=False)
    upFingers = detector.getUpFingers(frame)

    if upFingers:
        x, y = positions[8][0], positions[8][1]
        if upFingers[1] and not whiteBoard.isOver(x, y):
            px, py = 0, 0

            ##### pen sizes ######
            if not hidePenSizes:
                for pen in pens:
                    if pen.isOver(x, y):
                        brushSize = int(pen.text)
                        pen.alpha = 0
                    else:
                        pen.alpha = 0.5

            ####### chose a color for drawing #######
            if not hideColors:
                for cb in colors:
                    if cb.isOver(x, y):
                        color = cb.color
                        cb.alpha = 0
                    else:
                        cb.alpha = 0.5
            #Clear
        if clear.isOver(x, y):
            clear.alpha = 0
            canvas = np.zeros((720,1280,3), np.uint8)
        else:
            clear.alpha = 0.5

    # color button
    if colorsBtn.isOver(x, y) and not coolingCounter:
        coolingCounter = 10
        colorsBtn.alpha = 0
        hideColors = False if hideColors else True
        colorsBtn.text = 'Colors' if hideColors else 'Hide'
    else:
        colorsBtn.alpha = 0.5

    # Pen size button
    if penBtn.isOver(x, y) and not coolingCounter:
        coolingCounter = 10
        penBtn.alpha = 0
        hidePenSizes = False if hidePenSizes else True
        penBtn.text = 'Pen' if hidePenSizes else 'Hide'
    else:
        penBtn.alpha = 0.5



    #white board button
```

```
    if boardBtn.isOver(x, y) and not coolingCounter:
        coolingCounter = 10
        boardBtn.alpha = 0
        hideBoard = False if hideBoard else True
        boardBtn.text = 'Board' if hideBoard else 'Hide'

    else:
        boardBtn.alpha = 0.5




elif upFingers[1] and not upFingers[2]:
    if whiteBoard.isOver(x, y) and not hideBoard:
        #print('index finger is up')
        cv2.circle(frame, positions[8], brushSize, color,-1)
        #drawing on the canvas
        if px == 0 and py == 0:
            px, py = positions[8]
        if color == (0,0,0):
            cv2.line(canvas, (px,py), positions[8], color, eraserSize)
        else:
            cv2.line(canvas, (px,py), positions[8], color,brushSize)
        px, py = positions[8]

else:
    px, py = 0, 0
    # put colors button
    colorsBtn.drawRect(frame)
    cv2.rectangle(frame, (colorsBtn.x, colorsBtn.y), (colorsBtn.x +colorsBtn.w, colorsBtn.y+colorsBtn.h), (255,255,255), 2)

    # put white board buttin
    boardBtn.drawRect(frame)
    cv2.rectangle(frame, (boardBtn.x, boardBtn.y), (boardBtn.x +boardBtn.w, boardBtn.y+boardBtn.h), (255,255,255), 2)

    #put the white board on the frame
    if not hideBoard:
        whiteBoard.drawRect(frame)
        ########### moving the draw to the main image #########
        canvasGray = cv2.cvtColor(canvas, cv2.COLOR_BGR2GRAY)
        _, imgInv = cv2.threshold(canvasGray, 20, 255, cv2.THRESH_BINARY_INV)
        imgInv = cv2.cvtColor(imgInv, cv2.COLOR_GRAY2BGR)
        frame = cv2.bitwise_and(frame, imgInv)
        frame = cv2.bitwise_or(frame, canvas)



    ########## pen colors' boxes #########
    if not hideColors:
        for c in colors:
            c.drawRect(frame)
            cv2.rectangle(frame, (c.x, c.y), (c.x +c.w, c.y+c.h), (255,255,255), 2)

        clear.drawRect(frame)
        cv2.rectangle(frame, (clear.x, clear.y), (clear.x +clear.w, clear.y+clear.h),
(255,255,255), 2)


    ########## brush size boxes ######
    penBtn.color = color
    penBtn.drawRect(frame)
    cv2.rectangle(frame, (penBtn.x, penBtn.y), (penBtn.x +penBtn.w,
penBtn.y+penBtn.h), (255,255,255), 2)
    if not hidePenSizes:
        for pen in pens:
            pen.drawRect(frame)
            cv2.rectangle(frame, (pen.x, pen.y), (pen.x +pen.w, pen.y+pen.h), (255,255,255),
2)


    cv2.imshow('video', frame)
    #cv2.imshow('canvas', canvas)
    k= cv2.waitKey(1)
    if k == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```

HandTrackingModule.py

```
import cv2
import mediapipe as mp

class handDetector():
    def __init__(self, mode = False, maxHands = 2, modelComplexity = 1, detectionCon
= 0.5, trackCon = 0.5):
        self.mode = mode
        self.maxHands = maxHands
        self.modelComplex = modelComplexity
        self.detectionCon = detectionCon
        self.trackCon = trackCon

        self.mpHands = mp.solutions.hands
        self.hands = self.mpHands.Hands(self.mode, self.maxHands, self.modelComplex,
self.detectionCon, self.trackCon)
        self.mpDraw = mp.solutions.drawing_utils
        self.tipIds = [4, 8, 12, 16, 20]

    def findHands(self, img, draw = True):
        imgRGB = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        self.results = self.hands.process(imgRGB)
        if self.results.multi_hand_landmarks:
            for handLms in self.results.multi_hand_landmarks:
                if draw:
                    self.mpDraw.draw_landmarks(img, handLms,
self.mpHands.HAND_CONNECTIONS)
```

```
        return img

    def findPosition(self, img, handNo = 0, draw = True):
        self.lmList = []
        if self.results.multi_hand_landmarks:
            myHand = self.results.multi_hand_landmarks[handNo]
            for id, lm in enumerate(myHand.landmark):
                h, w, c = img.shape
                cx, cy = int(lm.x*w), int(lm.y*h)
                self.lmList.append([id, cx, cy])
                if draw:
                    cv2.circle(img, (cx,cy), 5, (255,0,0),cv2.FILLED)
        return self.lmList

    def fingersUp(self):
        fingers = []
        if self.lmList[self.tipIds[0]][1] < self.lmList[self.tipIds[0]-1][1]:
            fingers.append(1)
        else:
            fingers.append(0)
        for id in range(1,5):
            if self.lmList[self.tipIds[id]][2] < self.lmList[self.tipIds[id]-2][2]:
                fingers.append(1)
            else:
                fingers.append(0)
        return fingers
```

```
# def main():
#     cap = cv2.VideoCapture(0)
#     detector = handDetector()
#     while True:
#         success, img = cap.read()
#         img = cv2.flip(img,1)
#         img = detector.findHands(img)
#         lmList = detector.findPosition(img)
#         cv2.imshow("Image", img)
#         cv2.waitKey(1)
#
# def __name__ == "__main__":
#     main()
```

vwrite.py

```
import cv2
import numpy as np
import os
import HandTrackingModule as htm
from flask import Blueprint, render_template
from tensorflow.keras.models import load_model
import keyboard
import pygame
import time
```

```
VirtualPainter = Blueprint("HandTrackingModule", __name__, static_folder="static",template_folder="templates")

@VirtualPainter.route("/feature")
def strt():
    ############## Color Attributes ###############
    WHITE = (255, 255, 255)
    BLACK = (0,0,0)
    RED = (0,0,255)
    YELLOW = (0,255,255)
    GREEN = (0,255,0)
    BACKGROUND = (255,255,255)
    FORGROUND = (0,255,0)
    BORDER = (0,255,0)
    lastdrawColor = (0,0,1)
    drawColor = (0,0,255)
    BOUNDRYINC = 5

    ############## CV2 Attributes ###############
    cap = cv2.VideoCapture(0)
    width, height = 1280, 720
    cap.set(3, width)        #640, 1280
    cap.set(4, height)       #480, 720
    imgCanvas = np.zeros((height,width,3), np.uint8)


    ############## PyGame Attributes ###############
    pygame.init()
    FONT = pygame.font.SysFont('freesansbold.tff', 18)
    DISPLAYSURF = pygame.display.set_mode((width,
height),flags=pygame.HIDDEN)
    pygame.display.set_caption("Digit Board")
    number_xcord = []
    number_ycord = []

    ############## Header Files Attributes ###############
    folderPath = "header"
    myList = os.listdir(folderPath)
    overlayList = []

    for imPath in myList:
        image = cv2.imread(f'{folderPath}/{imPath}')
        overlayList.append(image)
    header = overlayList[0]

    ############## Predication Model Attributes ###############
    label=""
    PREDICT = "off"
    AlphaMODEL = load_model("bModel.h5")
    NumMODEL = load_model("bestmodel.h5")
    AlphaLABELS = { 0: 'a', 1: 'b', 2: 'c', 3: 'd', 4: 'e', 5: 'f', 6: 'g', 7: 'h', 8: 'i', 9: 'j',
    10: 'k', 11: 'l', 12: 'm', 13: 'n', 14: 'o', 15: 'p', 16: 'q', 17: 'r', 18: 's', 19: 't',
    20: 'u', 21: 'v', 22: 'w', 23: 'x', 24: 'y', 25: 'z', 26: ''}
    NumLABELS = {0:'0', 1: '1',
    2: '2', 3: '3',
    4: '4', 5: '5',
    6: '6', 7: '7',
    8: '8', 9: '9'}
    rect_min_x, rect_max_x = 0,0
    rect_min_y, rect_max_y = 0,0

    ############## HandDetection Attributes ###############
    detector = htm.handDetector(detectionCon=0.85)
    xp , yp = 0, 0
    brushThickness = 15
    eraserThickness = 30
    modeValue = "OFF"
    modeColor = RED

    while True:
        SUCCESS, img = cap.read()
        img = cv2.flip(img,1)

        img = detector.findHands(img)
        lmList = detector.findPosition(img, draw=False)
        cv2.putText(img,"Press A for Alphabate Recognisition Mode ",(0,145),3,0.5,(255,255,255),1,cv2.LINE_AA)
        cv2.putText(img,"Press N for Digit Recognisition Mode ",(0,162),3,0.5,(255,255,255),1,cv2.LINE_AA)
        cv2.putText(img,"Press O for Turn Off Recognisition Mode ",(0,179),3,0.5,(255,255,255),1,cv2.LINE_AA)
        cv2.putText(img,f'{"RECOGNISITION IS "}{modeValue}',(0,196),3,0.5,modeColor,1,cv2.LINE_AA)

        if keyboard.is_pressed('a'):
```

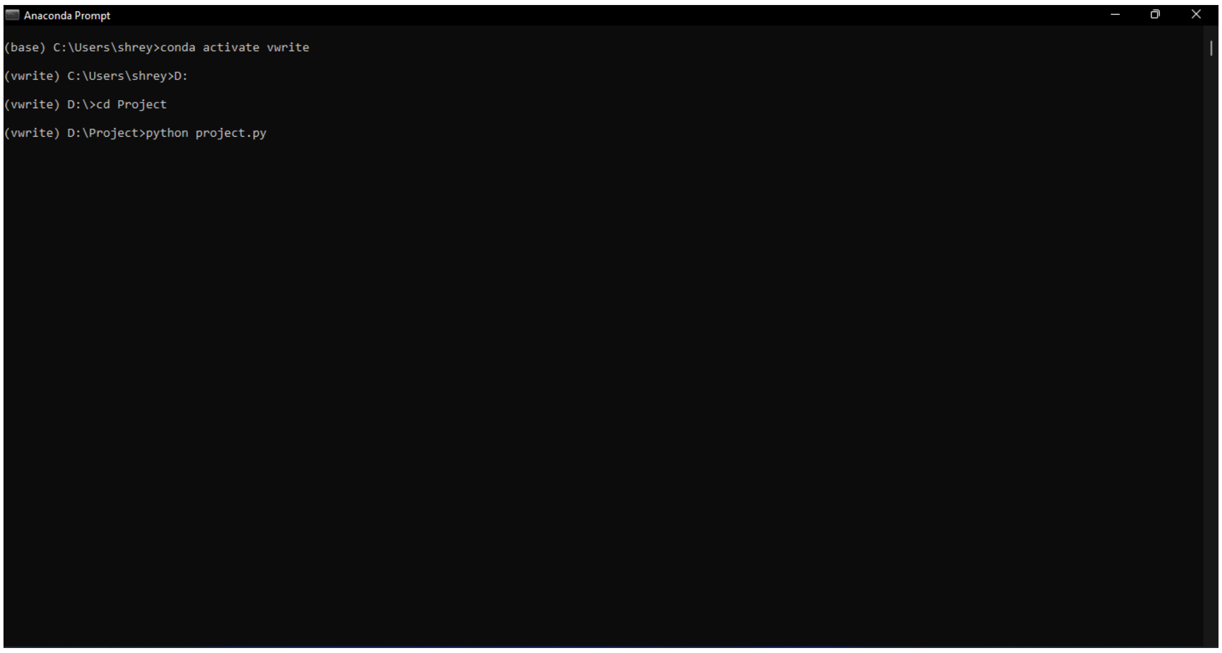
```
    if PREDICT!="alpha":
        PREDICT = "alpha"
        modeValue, modeColor = "ALPHABATES", GREEN

if keyboard.is_pressed('n'):
    if PREDICT!="num":
        PREDICT = "num"
        modeValue, modeColor = "NUMBER", YELLOW

if keyboard.is_pressed('o'):
    if PREDICT!="off":
        PREDICT = "off"
        modeValue, modeColor = "OFF", RED

    xp , yp = 0, 0
    label=""
    rect_min_x, rect_max_x = 0,0
    rect_min_y, rect_max_y = 0,0
    number_xcord = []
    number_ycord = []
    time.sleep(0.5)



if len(lmList)>0:

    x1,y1 = lmList[8][1:]
    x2,y2 = lmList[12][1:]
```

```
            fingers = detector.fingersUp()
            # print(fingers)

            if fingers[1] and fingers[2]:

                #add

                number_xcord = sorted(number_xcord)
                number_ycord = sorted(number_ycord)

                if(len(number_xcord) > 0 and len(number_ycord)>0 and PREDICT!="off"):
                    if drawColor!=(0,0,0) and lastdrawColor != (0,0,0):
                        rect_min_x, rect_max_x = max(number_xcord[0]-BOUNDRYINC, 0),
min(width, number_xcord[-1]+BOUNDRYINC)
                        rect_min_y, rect_max_y = max(0, number_ycord[0]-BOUNDRYINC),
min(number_ycord[-1]+BOUNDRYINC, height)
                        number_xcord = []
                        number_ycord = []

                        img_arr =
np.array(pygame.PixelArray(DISPLAYSURF))[rect_min_x:rect_max_x,rect_min_y:rec
t_max_y].T.astype(np.float32)


cv2.rectangle(imgCanvas,(rect_min_x,rect_min_y),(rect_max_x,rect_max_y),BORDER
,3)
                        image = cv2.resize(img_arr, (28,28))
                        # cv2.imshow("Tmp",image)
                        image = np.pad(image, (10,10), 'constant' , constant_values =0)
                        image = cv2.resize(image,(28,28))/255
                        # cv2.imshow("Tmp",image)

                        if PREDICT == "alpha":
                            label                                                                         =
str(AlphaLABELS[np.argmax(AlphaMODEL.predict(image.reshape(1,28,28,1)))])
                        if PREDICT == "num":
                            label                                                                         =
str(NumLABELS[np.argmax(NumMODEL.predict(image.reshape(1,28,28,1)))])
                        pygame.draw.rect(DISPLAYSURF,BLACK,(0,0,width,height))

                        cv2.rectangle(imgCanvas,(rect_min_x+50,rect_min_y-
20),(rect_min_x,rect_min_y),BACKGROUND,-1)
                        cv2.putText(imgCanvas,label,(rect_min_x,rect_min_y-
5),3,0.5,FORGROUND,1,cv2.LINE_AA)
                    else:
                        number_xcord = []
                        number_ycord = []

                xp, yp = 0, 0
                if y1<125:
                    lastdrawColor = drawColor
                    if 0 < x1 < 200:
                        imgCanvas = np.zeros((height,width,3), np.uint8)
                    elif 210 < x1 < 320:
                        header = overlayList[0]
                        drawColor = (0,0,255)
                    elif 370 < x1 < 470:
                        header = overlayList[1]
```

```
            drawColor = (0,255,255)
        elif 520 < x1 < 630:
            header = overlayList[2]
            drawColor = (0,255,0)
        elif 680 < x1 < 780:
            header = overlayList[3]
            drawColor = (255,0,0)
        elif 890 < x1 < 1100:
            header = overlayList[4]
            drawColor = (0,0,0)
        elif 1160 < x1 < 1250:
            cap.release()
            cv2.destroyAllWindows()
            quit()

    cv2.rectangle(img, (x1,y1-25), (x2,y2+25), drawColor, cv2.FILLED)



elif fingers[1] and fingers[2] == False:

    #add
    number_xcord.append(x1)
    number_ycord.append(y1)
    #addEnd



    cv2.circle(img, (x1,y1-15), 15, drawColor, cv2.FILLED)
    if xp == 0 and yp == 0:
```

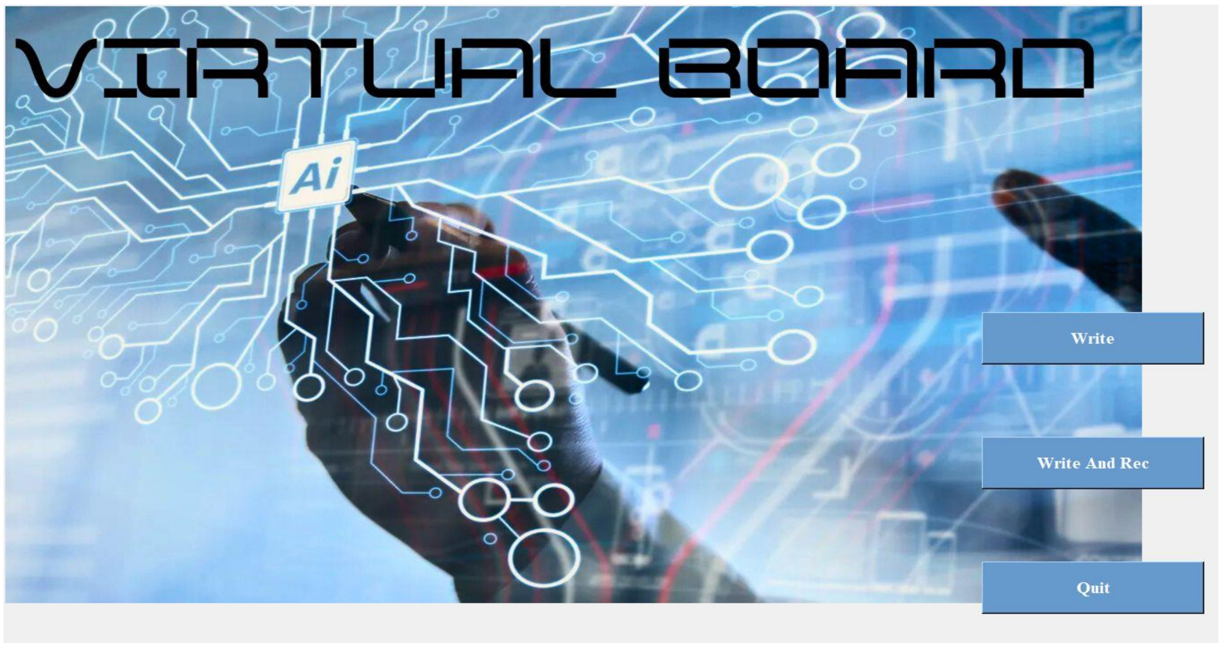
```
                xp, yp = x1, y1

            if drawColor == (0, 0, 0):
                cv2.line(img, (xp,yp), (x1,y1), drawColor, eraserThickness)
                cv2.line(imgCanvas, (xp,yp), (x1,y1), drawColor, eraserThickness)
            else:
                cv2.line(img, (xp,yp), (x1,y1), drawColor, brushThickness)
                cv2.line(imgCanvas, (xp,yp), (x1,y1), drawColor, brushThickness)
                pygame.draw.line(DISPLAYSURF,    WHITE,    (xp,yp),    (x1,y1),
brushThickness)
            xp, yp = x1, y1
        else:
            xp, yp = 0, 0

    imgGray = cv2.cvtColor(imgCanvas, cv2.COLOR_BGR2GRAY)
    _, imgInv = cv2.threshold(imgGray, 50, 255, cv2.THRESH_BINARY_INV)
    imgInv = cv2.cvtColor(imgInv, cv2.COLOR_GRAY2BGR)
    img = cv2.bitwise_and(img, imgInv)
    img = cv2.bitwise_or(img, imgCanvas)

    img[0:132,0:1280] = header
    pygame.display.update()
    # cv2.imshow("Paint",imgCanvas)
    cv2.imshow("Image",img)
    cv2.waitKey(1)

strt()
```

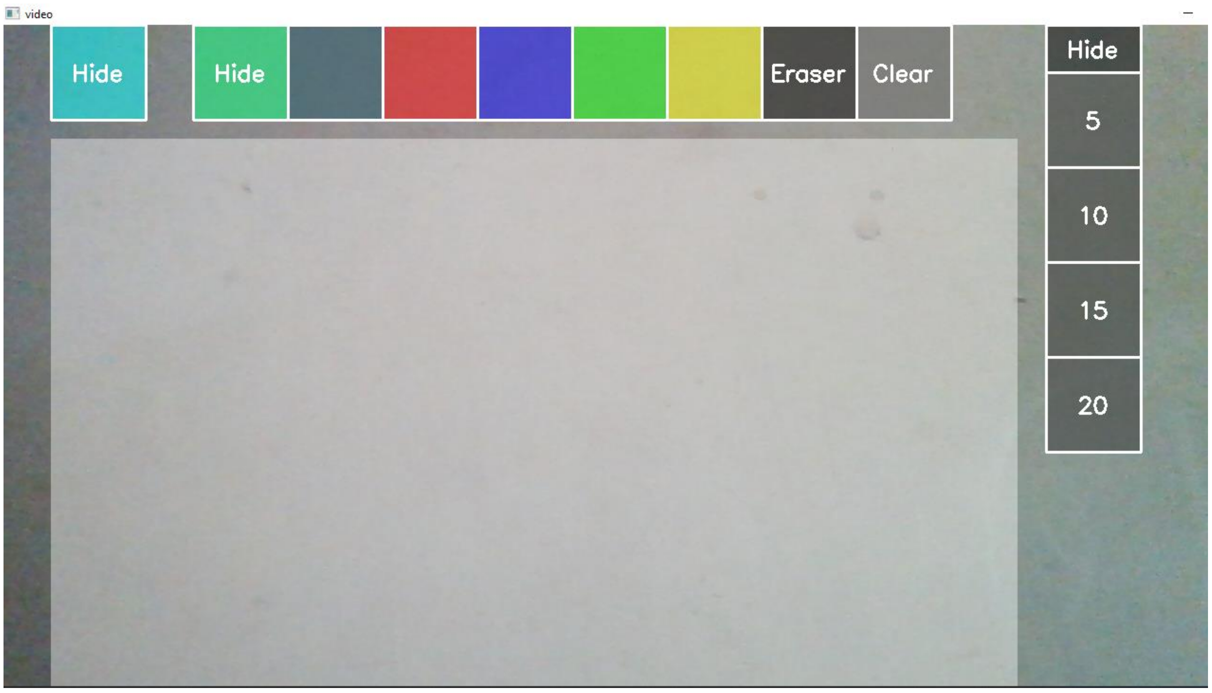
11.2 INPUT SCREEN DESIGN

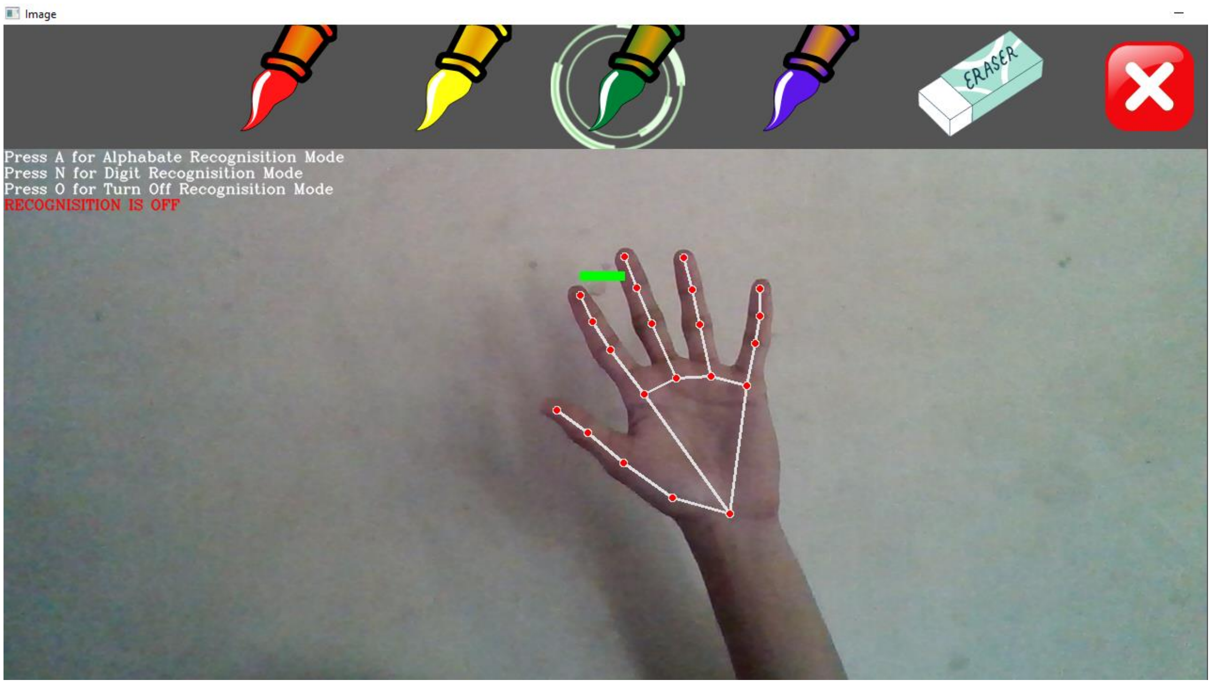


Anaconda Prompt

```
(base) C:\Users\shrey>conda activate vwrite

(vwrite) C:\Users\shrey>D:

(vwrite) D:\>cd Project

(vwrite) D:\Project>python project.py
```

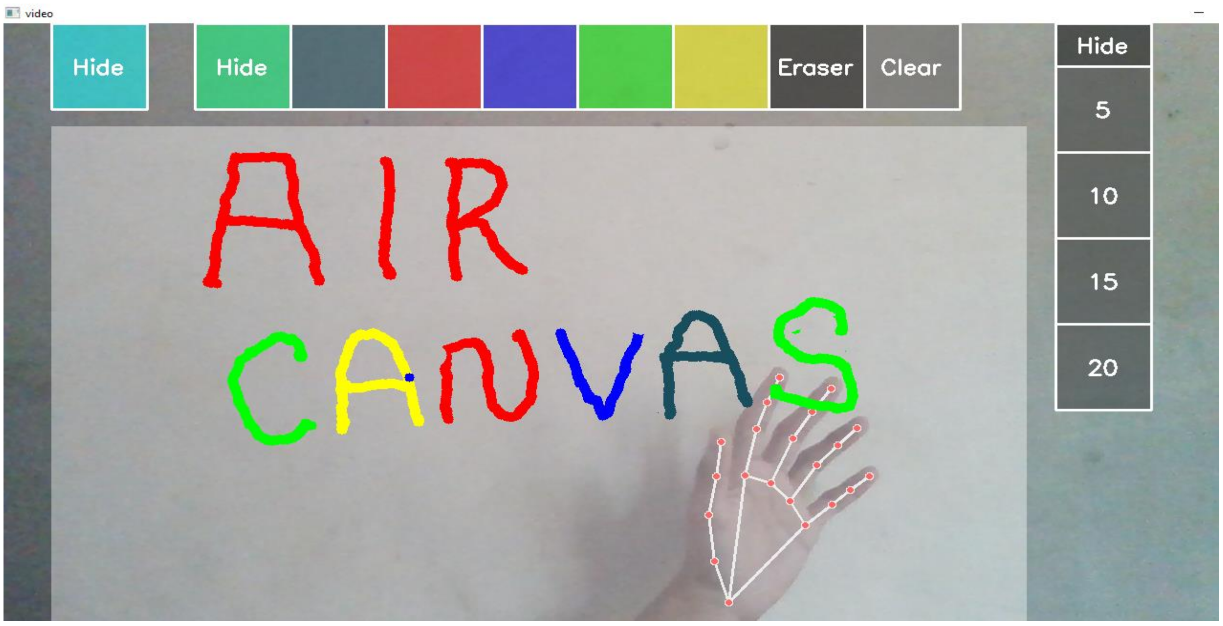


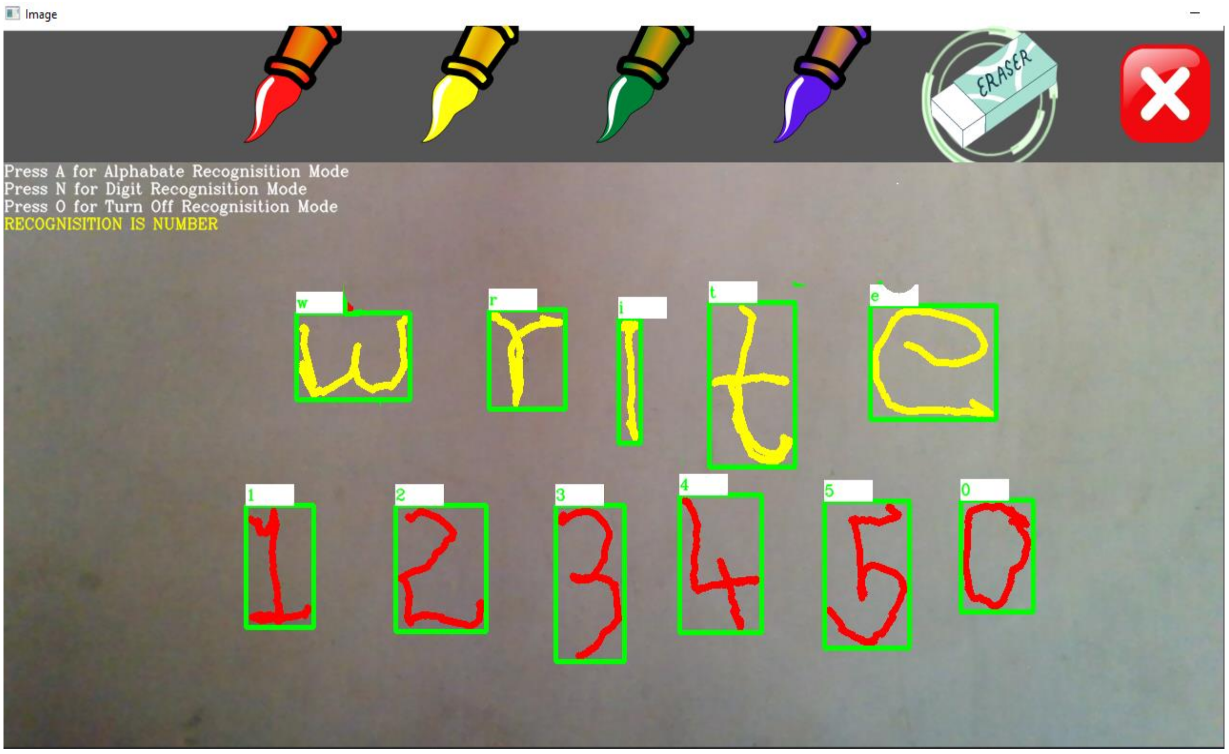




```
Press A for Alphabate Recognisition Mode
Press N for Digit Recognisition Mode
Press O for Turn Off Recognisition Mode
RECOGNISITION IS OFF
```

11.2 OUTPUT SCREEN DESIGN





Press A for Alphabate Recognisition Mode
Press N for Digit Recognisition Mode
Press O for Turn Off Recognisition Mode
RECOGNISITION IS NUMBER