Abstract

Character Recognition involves Electronic Detection and Identification of Printed or Handwritten Text in an Image. For a Finger to be Used as a Tool for Writing, it needs to be Detected, Recognized and Tracked by the Algorithm. There are Certain Issues with Human Handwriting. The Ambiguity of Human Handwriting and the Variability of the Finger Detection are to name a few. Machine Learning can be used to Recognize Human Fingers, and can also detect Patterns from the Strokes, allowing us to Extract Handwritten Text.

Acknowledgement

With this Project, we wish to thank our internal guides **Dr. Bhaskar Bhuyan** and **Dr. Palash Goshal** who were of constant guidance throughout this journey. We also thank the entire Information Technology department to let us create projects that further enrich our curricular as well as co-curricular aspects of project development. And lastly, we thank all the people who we have met during this journey of 'Smart AirPad' that have been of support and assistance.

Aryamaan Borah (201900049)

Biswadeep Sarkar (201900322)

Index

1. Introduction

Languages and Communication have existed since time immemorial. Initially, communication involved glyphs and images, which were carved on walls and stone slates. As time passed, the glyphs started representing language units to form rules called grammar. As text became extensive, the units became simpler and shorter to represent the today's generation of language scripts that allow us to use written text for communication.

Written text will require a certain medium to write on it, and a writing tool to write with it. The most common medium for writing is paper, and various tools such as pens and pencils are used. While pen and paper are still the easiest to acquire, touch screen input and finger strokes became a viable choice for text inputs and annotations, but were still contact based.

Finger strokes in air (virtually), combined with machine learning for finger detection and text recognition, gives us Air Writing. Air Writing involves gesture based, human to computer interaction in three-dimensional space. Although air writing involves 3D space in real-time, cameras are often unable to perceive depth of a 3D Gesture accurately.

The six degrees of freedom in air writing, combined with the overall variability of handwritten text, makes text recognition in air writing, a challenging task.

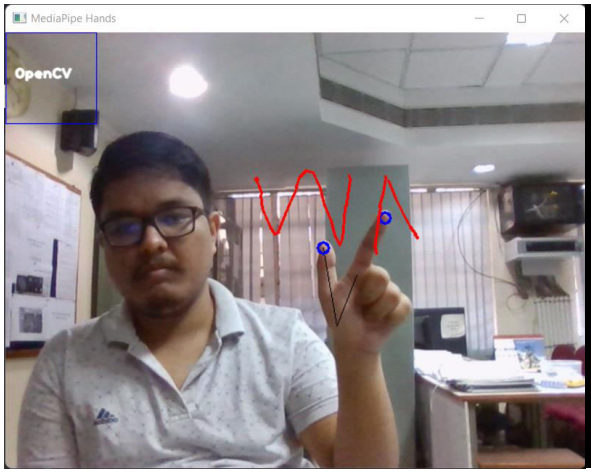


Figure 1: Actual Video Frame

Figure 2: Segmented Strokes

1.1 General Overview of the Problem

There are a variety of input devices that are available for use. But the problem arises when we wish to deal with handwritten text as an input for a computer. Styli and trackpads exist, but are uncommon for the general public. This also contributes to hardware-based limitations, where the user must have these peripherals in their vicinity to communicate with handwritten text.

This project aims to overcome these limitations by implementing air writing on a camera feed and detecting text from the feed. This serves as a more natural input mechanism for the general public and is not limited by specialized hardware. This project requires only a monitor and a webcam as the end user peripherals for handwritten text input.

Along with the handwritten text, this program will also be capable of recognising the text from the image and perform annotations and save them to disk.

1.2 Literature Survey

Author and Publication	Title of the Paper	Findings	Relevance to the Project
1. Yilmaz, A., Javed, O. and Shah, M., 2006. Object tracking: A survey. *Acm computing surveys (CSUR)*, *38*(4), pp.13-es.	Object Tracking: A Survey	The Finger is Tracked by placing an LED on the Finger.	By Making the Finger Tip visually Different from the Background, Fingertip Detection can be Simpler.
2. Agrawal, A., Raj, R. and Porwal, S., 2013, April. Vision-based multimodal human-computer interaction using hand and head gestures. In 2013 IEEE Conference on Information & Communication Technologies (pp. 1288-1292). IEEE.	Vision-based Multimodal Human-Computer Interaction using Hand and Head Gestures	From a set of positive and negative image samples, the AdaBoost Algorithm can select the best weak Classifier.	Viola-James Algorithm and Haar Features can be used for Hand Detection.
3. Baig, F., Fahad Khan, M. and Beg, S., 2013. Text writing in the air. Journal of Information Display, 14(4), pp.137-148.	Text Writing in Air	Infrared Light is Projected towards the User and Reflected Infrared is Captured, making a Depth Map	Kinect Sensor can perform Edge Detection of Hands by using the Depth Maps
4. Alam, M., Kwon, K.C., Abbass, M.Y., Imtiaz, S.M. and Kim, N., 2020. Sensors, 20(2), p.376.	Trajectory-Based Air-Writing Recognition Using Deep Neural Network and Depth Sensor	Depth Cameras have been used to provide additional 3D information.	Hand Trajectory with Depth leads to online features of text which can be further simplified with CNN and LSTM
5. Ptucha, R., Such, F.P., Pillai, S., Brockler, F., Singh, V. and Hutkowski, P., 2019. Intelligent character recognition using fully convolutional neural networks. Pattern recognition, 88, pp.604-613.	Intelligent character recognition using fully convolutional neural networks	Input Image is divided into regions using filters.	Instead of collectively examining pixels of text, it is possible to check regions of the image, where each region is compared to a pre-determined symbol using CNN.

Table 1: Literature Survey

1.3 Analysis of the Problem

Air Writing involves using strokes in air virtually, and with such strokes we write text with it. This can lead to certain problems.

The hand and its corresponding gestures need to be detected. The finger tips behave as a writing tool and the air behaves as the medium. The hand and its finger tips must be tracked in real-time to get the position of the pixels of the strokes. The hand gestures must also be validated as such gestures are innumerable and any random movement of the hand should not produce a stroke. With this in mind, it may be possible to assign certain 'air-writing feature' to a specific hand gesture.

The text that is generated with strokes will not be natural (like handwritten text) but will also not be uniform (like machine text).

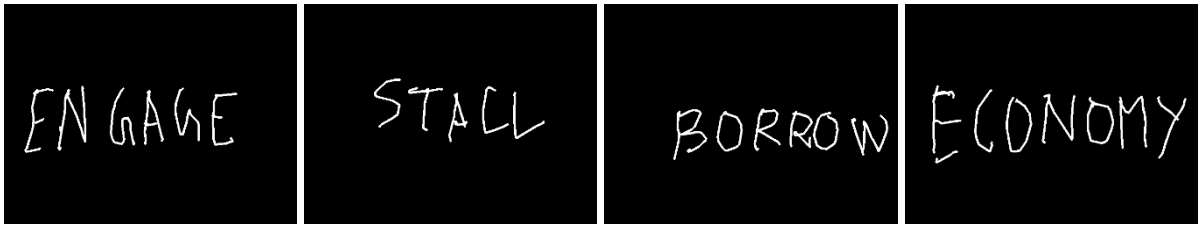


Figure 3: Multi-stroke text generated during air writing

This implies that any text recognition model for this scenario will not generate accurate results directly. For this case we will require pre-processing on the image text first, and then the text recognition models will become applicable. After the text is processed the recognized text is sent back to the user.

Utilities such as clearing the screen will also be provided to the user via the camera feed. These utilities can be performed by the finger strokes as well.

2. Software Requirements Specification (SRS) document

2.1 Introduction

Purpose

The purpose of this document is to implement a concept for text input via finger strokes in air.

Intended Audience and Intended Use

This project is a concept that involves text and is usable by anyone who has effective understanding of English characters and words. However, certain categories of people (such as mute/deaf people) may have a stronger advantage with its implementation.

The intended use involves detecting and recognizing english text input and is expected to be used as a contact-free electronic text input mechanism.

Project Scope

This concept makes use of a video camera and can detect hands and fingers. It aims to generate text from strokes and recognize the text from these strokes. As for now, the project aims to deal with only English text.

2.2 Overview

Project Features

- This project makes use of a video camera to detect hands and fingers.
- Finger strokes will generate annotations on the camera feed that will be visible on a digital display.
- The program involves a text detection and recognition algorithm that can obtain the text from the annotations.
- Annotations along with the camera feed can be stored in secondary storage.
- A contact-free interface for drawing on the feed, cleaning the strokes from the screen, storing the annotations and other basic functions will also be provided.

User Needs

- Text detection and recognition using finger strokes can be an effective input mechanism that does not involve input peripherals.
- Free hand annotations (such as shapes) can also be drawn to better enhance the context.
- Users with disabilities (mute/deaf people) can make effective use of this concept.
- Video camera feed can often be incorporated with long ranged communications, which can be vital at times.

2.3 Product Requirements

Functional Requirements

- The video camera records the image and the hand detection and recognition algorithm starts working on it.
- Upon successful detection of a hand, the algorithm generates the coordinates of the finger tips and finger joints with respect to a coordinate system of the recorded image.
- The index finger behaves as the drawing point for the annotations. When the thumb is brought near the index finger, the drawing begins. Conversely, the drawing stops when the thumb is taken away from the index finger.
- The video camera works on real-time, and records multiple images in a second, leading to a continuous curve of recorded points, leading to a drawing stroke.
- A digital display shows the user the video feed and the strokes that were drawn.
- Upon interacting with the interface with the index finger, the text detection and recognition algorithm starts working on valid text from the annotations, if any.
- Recognized text will be displayed on the digital display for the user.
- The interface also provides other features such as clearing the screen and storing the annotations.

Non-Functional Requirements

- Video camera resolution must be at least 640x480. Higher resolutions are possible, but the exact resolutions are camera and driver software specific.
- Higher quality video cameras can have better focus and aperture, which could remove latency and improve image quality. This is optional.
- The digital display needs to be of adequate size to support the resolution of the video camera feed.
- It is recommended to make use of a background that does not interfere with skin color for the hand detection module.

2.4 Interface Requirements

Hardware Requirements

- Video Camera
- Digital Display Device
- GPU based Processor
- Secondary Storage Devices (Optional for Storing Images)
- 16 GB RAM

Software Requirements

- Any Modern Operating System (to Execute the Program)

3. Design

Software Development Lifecycle Model (SDLC) – RAD:

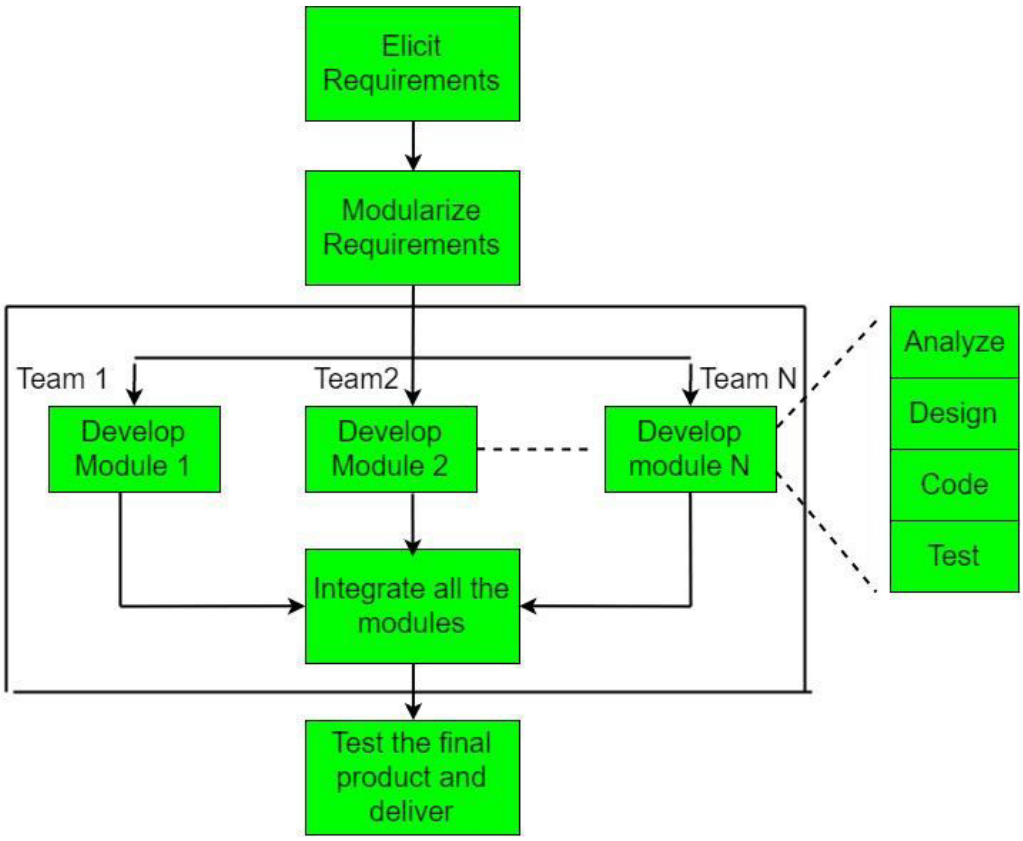
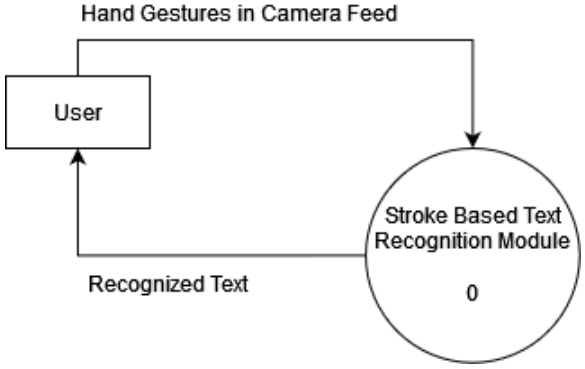
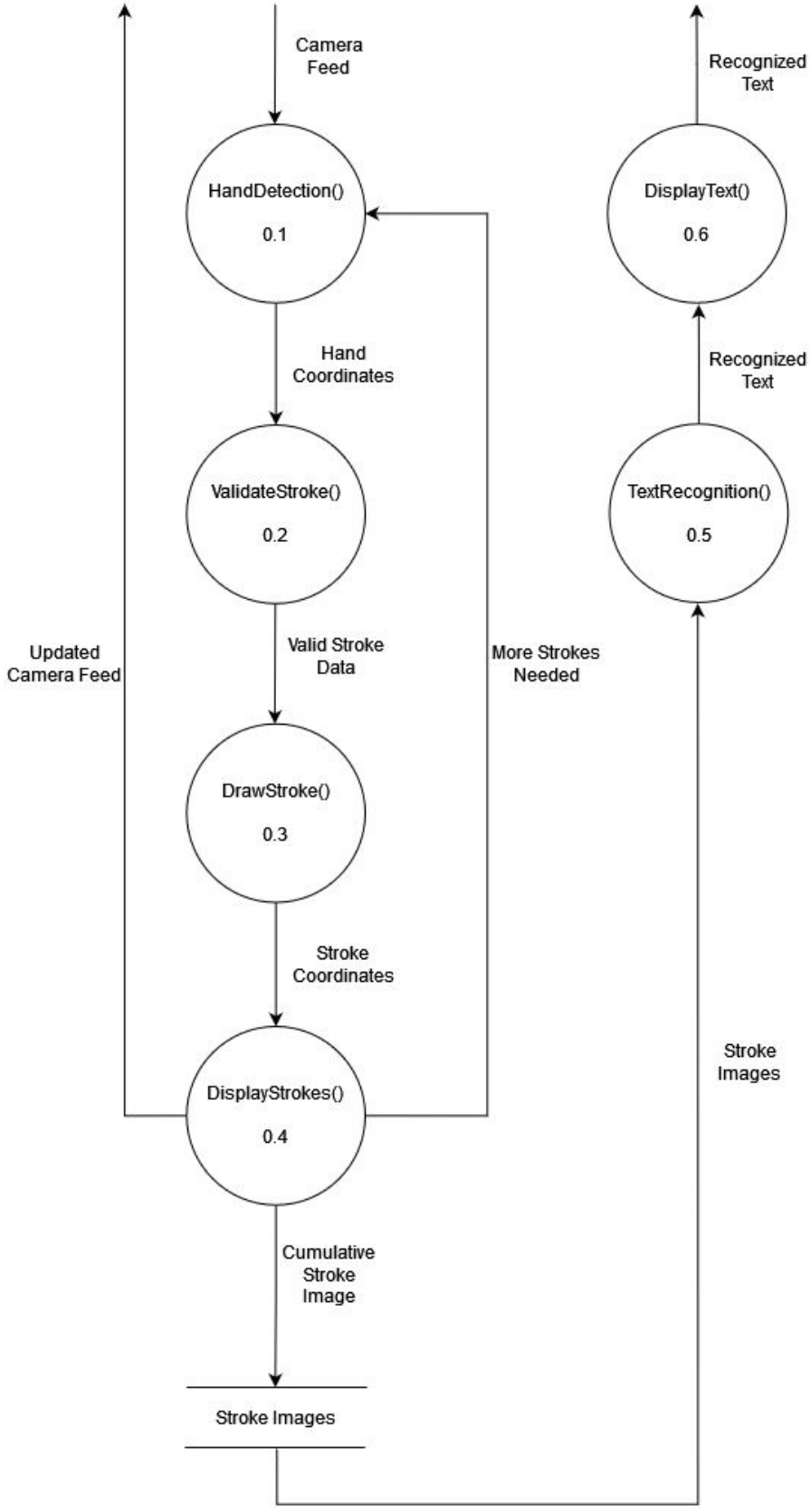


Figure 4: Rapid Application Development (RAD) model

Context Diagram (Level 0):



Data Flow Diagram (Level 1):



Data Dictionary:

Camera Feed	Video Camera Feed. It consists of a stream of BGR Images where the most recent camera feed frame is made available for use.
Hand Coordinates	X-Y Coordinates generated after a successful hand detection. These coordinates point to the fingertips and joints of the recognized hand in the image and are generated with respect to the wrist, where the wrist is (0, 0). Hand Coordinates are initially generated in floating points, but are converted to integer for actual usage.
Valid Stroke Data	X-Y Stroke Coordinates. The fingertips of the recognized hand can be used to generate strokes, but only a specific hand gesture results in actual stroke generation. The valid gesture, combined with the coordinates of the fingertips, results in an array of X-Y Stroke Coordinates.
Stroke Coordinates	The finalized X-Y Stroke Coordinates. These are valid coordinates that can be used to draw lines with, resulting in a continuous stroke.
Updated Camera Feed	The stroke coordinates are joined using lines and are drawn directly on the camera feed, letting the user see the strokes they have created.
Cumulative Stroke Image	Multiple non-connecting strokes can be generated on the same image, leading to text generation. These stroke images can then be stored in the disk.
Recognized Text	The text recognition module works on these stroke images to generate the text in the image. Text is expressed as character strings.

Table 2: Data Dictionary

4. Implementation Details

OpenCV with Python allows us to extract individual frames from a webcam. MediaPipe works side-by-side with Python to detect hands from the extracted webcam frames, generating floating point coordinates relative to the wrist of the detected hand. These coordinates are converted to absolute integer coordinates where the upper-left corner of the image is (0, 0).

For hand gestures, we have decided to take the angle between the thumb and index finger for stroke validity.

$$\tan\theta = \frac{m_1 - m_2}{1 + m_1 m_2}$$

Here, we deal with four coordinates of the hand, the thumb tip (TT), the index finger tip (IT), the index finger joint (IJ) and the thumb joint (TJ). TT and TJ make a line with a slope m_1 and IJ and TJ make the other line with slope m_2, letting us obtain the angle between these two lines. Experimentally, the value of tan θ is effective within the range of (-0.3, 0.3). Whenever tan θ is within this range, the coordinates of the index finger tip (IT) are recorded for stroke drawing. This allows us to obtain an array of valid stroke coordinates.

We take two consecutive stroke coordinates from the array and draw a line between them. This is repeated for all the coordinates in the array to obtain the final stroke. This stroke is drawn on the camera feed and on a blank black image. The former allows the user to see the stroke they have drawn, and the latter allows the stroke to be segmented into a blank image for further processing, and is saved on disk. This process can be repeated multiple times for the same image to generate multiple strokes, as most English characters are multi-stroke characters.

The segmentation process involves drawing the stroke with the stroke coordinates obtained in the previous step. We have decided to use the segmented image for implementing the text recognition machine learning model and for image dataset generation. For text recognition we have used the Multi-Object Rectified Attention Network (MORAN) model.

MORAN makes use of two Neural Networks: Multi-Object Rectification Network (MORN) and Attention-Based Sequence Recognition Network (ASRN).

Rectification Networks (for text) work on modifying irregular text into regular text. Irregular text involves text that is visually transformed, making it difficult for existing text recognition models to work accurately. Such networks are capable to rectify such irregularities in text to a certain extent. MORN processes the segmented image using a Convolutional Neural Network (CNN) where various regions in the image are shifted in horizontal and vertical directions by adding or subtracting pixels in the required directions to straighten the text. This also leads to

a boundary box rectification. Multi-Object in MORN refers to separate but interrelated objects being used for rectification.

Recognition Networks are based on pattern recognition. These networks can identify existing patterns, assign values to them and perform comparisons between the patterns. ASRN makes use of Recurrent Neural Networks (RNN) to decode the text. Attention-Based in ASRN implies that only specific regions in the image need to be focussed to extract the text.

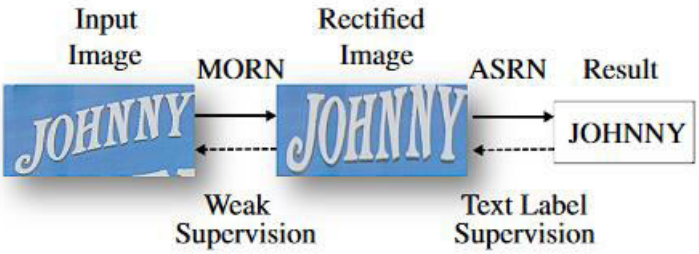


Figure 5: Basic Working Principle of MORAN

5. Results

MediaPipe has a 95.7% in hand (palm) detection and has a lower limit of just 86.22%. This is in accordance to the MediaPipe documentation as offered by the Google Development Team. Link: https://google.github.io/mediapipe/solutions/hands.html

The Pre-Trained MORAN model upon testing it with 100 segmented images obtained from the program results in an accuracy of 30.0%. This has encouraged us to use manually procured datasets and train the model accordingly for better accuracy.

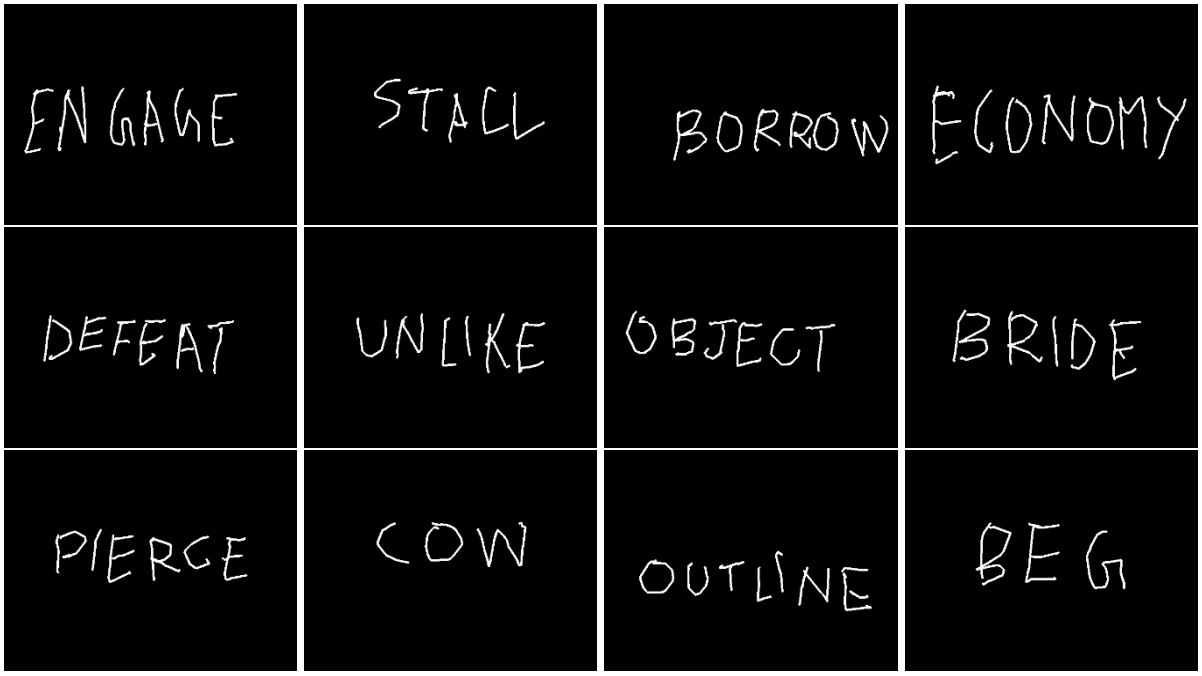


Figure 6: Segmented Images for Datasets

6. Summary and Conclusions

6.1 Summary of Achievements

- Incorporated a pre-trained hand detection model.
- Implemented a stroke validation mechanism for recording valid strokes.
- Implemented an interface for clearing the camera feed.
- Implemented an interface for storing the images.
- Implemented a pre-trained CNN model for regularization of irregular text in an image (MORN).
- Implemented a pre-trained RNN model for recognition of text in an image (ASRN).
- Displayed recognized text to the user.

6.2 Major Difficulties

- Pre-Trained Character Recognition Models do not work effectively, as Stroke Based Text looks Substantially Different than Handwritten Text. For this case we require a Manually Trained Model with Image Datasets, along with experimentation with existing models.
- Image Datasets are much harder to collect, because of their size and procurement difficulty.

6.3 Limitations

- The hand detection algorithm might fail if the hands are positioned too close or too far away from the camera, or if the complete hand is not visible in the camera feed.

- The hand detection algorithm might fail if the background of the camera feed is the same color as that of the input hand.

6.4 Future Scope of the Work

As of today's generation, there exists a variety of input devices made available to the general public. This also involves the common keyboard which can be used for generating machine text and the mouse to generate basic 2D movement via dragging. As for drawing and writing handwritten text, there exists styli and joysticks. But there still exists two limitations. These input devices are contact based and are hardware intensive.

By creating an input mechanism that does not involve input devices, it can greatly simplify communication, especially when used in short messages. This can save money on input devices, and can also be used as a public communication mechanism as there are very few peripherals involved.

It allows deaf and mute people to indulge in conversations without dealing with sign language. This also negates the use of sign language translators. The lack of writing medium is also helpful. This can save on paper and trees, as paper is still one of the preferred mediums of writing.

7. Gantt Chart

ACTIVITIES	Feb 2022				March 2022				April 2022				May 2022				June 2022	
	WEEK 1	WEEK 2	WEEK 3	WEEK 4	WEEK 1	WEEK 2	WEEK 3	WEEK 4	WEEK 1	WEEK 2	WEEK 3	WEEK 4	WEEK 1	WEEK 2	WEEK 3	WEEK 4	WEEK 1	WEEK 2
Literature Review																		
Researching																		
Logic and Design																		
Coding																		
Testing and Debugging																		
Documentation																		
COLOR CODES:																		
Expected to be done																		
Completed																		

8. Conclusions

The project implements stroke-based text input via air writing. The hand detection algorithm implemented using MediaPipe was of high accuracy (95.7%) and we were able to successfully segment the strokes of the camera feed onto a blank image for further processing, or let the user store them as annotations, without any processing.

The text recognition model which used Multi-Object Rectified Attention Network (MORAN) was of low accuracy (30.0%). We are still continually trying to enhance the accuracy by using a larger dataset and training it manually.

9. Bibliography

[1] Yilmaz, A., Javed, O., & Shah, M. (2006). Object tracking: A survey. Acm computing surveys (CSUR), 38(4), 13-es.

[2] Agrawal, A., Raj, R., & Porwal, S. (2013, April). Vision-based multimodal human-computer interaction using hand and head gestures. In 2013 IEEE Conference on Information & Communication Technologies (pp. 1288-1292). IEEE.

[3] Baig, F., Fahad Khan, M., & Beg, S. (2013). Text writing in the air. *Journal of Information Display*, *14*(4), 137-148.

[4] Alam, M., Kwon, K.C., Abbass, M.Y., Imtiaz, S.M. and Kim, N., 2020. Trajectory-based air-writing recognition using deep neural network and depth sensor. *Sensors*, *20*(2), p.376.

[5] Ptucha, R., Such, F.P., Pillai, S., Brockler, F., Singh, V. and Hutkowski, P., 2019. Intelligent character recognition using fully convolutional neural networks. Pattern recognition, 88, pp.604-613.