International Journal of Engineering Applied Sciences and Technology, 2024
Vol. 9, Issue 04, ISSN No. 2455-2143, Pages 95-99
Published Online August 2024 in IJEAST (http://www.ijeast.com)

MODEL IMPLEMENTATION AND COMPREHENSIVE STUDY ON VISUAL AIR TYPING SYSTEMS

Adarsh Jha, Bhavisha Narendra Chaudhari
Department of Computational Intelligence
SRM Institute of Science and Technology, Chennai,
India

Abstract—**This paper explores the implementation and applications of visual air typing systems, focusing on fingertip detection and text recognition to provide an alternative to the traditional input methods. Although models with similar functionality exist, our study emphasizes how such systems can offer practical solutions for users with visual impairments and enhance user experience in virtual reality (VR) environments. By detecting fingertip movements and translating them into written words, such models present a seamless and intuitive user experience. For individuals with visual impairments, traditional keyboards can be challenging, and sign language lacks uniformity across different regions. An air typing system addresses these issues by offering a consistent text input method. Additionally, in VR applications where traditional keyboards are impractical, it enables more natural and efficient text entry. This paper provides a comprehensive review of existing air typing technologies, detailing their strengths and limitations. We discuss the potential future directions for air typing systems, highlighting the need for more inclusive and versatile input methods. Our findings indicate that air typing can significantly enhance user interaction in diverse environments, making it a valuable alternative to traditional keyboards. Overall, this study underscores the practical applications and benefits of air typing systems, particularly for visually impaired users and VR contexts, contributing to advancing more inclusive technological solutions.**

Keywords—**Air Typing, Keyboard, virtual reality, input methods**

I. INTRODUCTION

The emergence of Computer Vision and Machine Learning technologies has opened many opportunities for the advancement of Human Computer Interaction (HCI), such as air writing and gesture recognition. Air writing enables users to write mid-air using gestures and use this as a form of input or communication with the computer. This technology heavily relies on the advancements in hand tracking and pattern recognition, particularly using frameworks like MediaPipe and OpenCV,**Error! Reference source not found.**[1] to identify and interpret human gestures in real-time. The easy accessibility of these technologies has made this field a major area of research and development in HCI.

The fingertip tracking capability provided by MediaPipe is essential for accurately capturing the fine movements required for air writing. By focusing on the fingertip, the system can determine the exact path traced by the user's hand, which is crucial for correctly interpreting the intended characters or symbols. This approach minimizes errors and ensures that the captured gesture closely matches the user's intended input. The use of MediaPipe also offers the advantage of being lightweight and efficient, making it suitable for real-time applications where responsiveness is critical.

Once the gesture is captured, the Google Vision API[3] is employed to interpret the air-written characters or shapes. The API's robust image recognition capabilities allow it to accurately detect and classify the input, converting the hand gestures into recognizable text or symbols. This combination of fingertip tracking and advanced image recognition creates a seamless air writing experience, where users can write or draw in the air with high accuracy and minimal latency. The integration of these technologies not only enhances the usability of air writing systems but also broadens their potential applications in areas such as virtual reality, assistive technology, and interactive displays.

Our work constitutes a simple yet effective implementation approach that would enhance the accessibility and reliability of air writing technology. By effectively integrating the capabilities of MediaPipe and the Google Vision API, our system successfully addresses and transcends many of the limitations inherent in earlier methods that depended solely on traditional image processing techniques. This innovative approach not only increases the precision of air writing recognition but also opens avenues for more intuitive and user-friendly human-computer interactions.

II. RELATED WORK

In recent years, significant advancements in deep learning and artificial intelligence have driven research into real-time

gesture recognition systems, particularly for American Sign Language (ASL)[4] recognition. Gesture recognition plays a crucial role in improving communication between hearing and hearing-impaired individuals by translating hand movements into text or spoken language. One of the approaches, as discussed by Taskiran et al.**Error! Reference source not found.**, employs Convolutional Neural Networks (CNNs) to classify hand gestures based on image data. Their study utilized a dataset from Massey University and achieved remarkable test accuracy. The researchers focused on extracting hand gestures using convex hull algorithms and real-time skin detection, achieving 98.05% accuracy in recognizing ASL gestures through a real-time system.

Similarly, another recent study by Ba and Bagyammal[5] utilized Google's MediaPipe framework and Long Short-Term Memory (LSTM)[6] models to enhance the accuracy of gesture recognition. MediaPipe, which is effective in detecting 3D hand landmarks, was employed for the recognition of both static and dynamic hand gestures. The LSTM model allowed the system to capture sequences of hand movements, achieving a high accuracy rate of 99% for ASL alphabet recognition. Their dataset included permutations of hand gestures from different individuals, enhancing the robustness of their model.

Both of these approaches showcase the potential of machine learning techniques like CNNs and LSTMs in improving gesture recognition systems. By leveraging real-time hand tracking and deep learning-based classification methods, these studies contribute significantly to bridging the communication gap for hearing-impaired individuals. The high accuracies achieved in both systems highlight the effectiveness of using advanced neural network architectures for ASL recognition, making these models highly applicable in real-world environments.

While American Sign Language (ASL) is one of the most widely recognized sign languages, it is important to note that there are numerous other sign languages globally, each with unique grammar, syntax, and gestures. Examples include British Sign Language (BSL), French Sign Language (LSF), and Indian Sign Language (ISL), among others. The approaches discussed above, such as the CNN-based recognition system by Taskiran et al. and the LSTM-based model by Ba and Bagyammal, are specifically tailored to ASL. Since these systems are trained on ASL datasets, they rely heavily on the distinct hand shapes and movements used in ASL. Consequently, their effectiveness would diminish when applied to other sign languages, as the gestures and structures in BSL, LSF, or ISL may differ significantly. Without retraining or adaptation to the specific features of these other sign languages, the current models would likely fail to provide accurate recognition, limiting their generalizability across sign language systems.

In addition, there is a necessity for an individual to memorise all these signs accurately in order for them to use this as an alternate form of input method. Therefore, air writing systems provide a better solution by overcoming such limitations. Air-writing systems, which allow users to write characters or words in free space using hand or finger movements, have become increasingly relevant for human-computer interaction, especially in applications where traditional input methods are impractical.

Chen et al. (2016)[7] focus on air-writing recognition using six-degree-of-freedom (6-DOF) motion data captured via optical and inertial sensors. This method models individual letters as "motion characters" and connects them into "motion words" using ligature models. To handle the variability in motion gestures, they employ Hidden Markov Models (HMMs) to model the writing process. Their system demonstrates a word error rate of 0.8% for word-based recognition and 1.9% for letter-based recognition, highlighting its effectiveness in recognizing connected and overlapped writing.

Hsieh et al. (2021)[8] present a deep learning-based air-writing recognition system that leverages convolutional neural networks (CNNs). Their approach captures air-writing trajectories using a 2D camera, allowing users to write freely without requiring an imaginary box or delimiter. This system preprocesses the x and y coordinates of the user's hand movement into 1D or 2D arrays, significantly reducing the complexity of the neural network while maintaining high accuracy. Their method achieves real-time recognition with an accuracy of over 99%, outperforming traditional image-based recognition methods.

Both approaches contribute to the advancement of air-writing systems by addressing fundamental recognition challenges. Chen et al. (2016) provide deep insights into the complexities of free-space writing, particularly for continuous and overlapped letters, while Hsieh et al. (2021) offer a CNN-based system that excels in simplicity and real-time application. These methodologies are critical for improving air-writing recognition systems, which can be applied in various hands-free environments, such as smart homes, virtual reality, and intelligent system control.

Multiple machine learning models have been employed to optimize hand gesture recognition systems. Mishra et al. (2023)[9]experimented with models like Dense Networks, CNN, and KNN, concluding that CNN provided the highest accuracy (99.21%) at the expense of longer training times. The researchers also found that MediaPipe, combined with dense models, significantly reduces training time while maintaining high accuracy. Quiñonez et al. (2022)[10] further demonstrated the adaptability of MediaPipe by integrating it with other machine learning platforms, such as TensorFlow, and applying it in various use cases like face and hand tracking.

In summary, both Mishra et al. (2023) and Quiñonez et al. (2022) underscore the effectiveness of MediaPipe as a tool for gesture recognition, particularly when combined with machine learning models. Its ability to perform real-time tracking with minimal computational resources makes it a compelling

International Journal of Engineering Applied Sciences and Technology, 2024
Vol. 9, Issue 04, ISSN No. 2455-2143, Pages 95-99
Published Online August 2024 in IJEAST (http://www.ijeast.com)

choice for implementing gesture-controlled systems in smart home environments.

III.METHODOLOGY

A. Fingertip Detection –

The first step in the Visual Air Typing System involves detecting fingertip and hand landmarks using the Mediapipe Hands model, a machine learning framework capable of identifying 21 key points on each hand, including the fingertips, knuckles, and wrist as shown in Fig. 1. The model processes individual video frames in real-time and provides the normalized 3D coordinates (x, y, z) of these landmarks, where (x) and (y) are normalized to the frame dimensions (ranging from 0 to 1), and (z) represents the depth relative to the wrist. This detailed spatial information allows for the accurate tracking of each finger's position. The fingertip detection algorithm leverages this data to precisely monitor fingertip movements, forming the foundation for interpreting complex gestures. The system's ability to process this information in real-time ensures a seamless and responsive experience, enabling efficient translation of air typing gestures into text inputs, even during rapid hand motions. The fingertip detection algorithm can be described as follows:

Algorithm: Fingertip Detection using Mediapipe Hands
Input: Video stream from camera
Output: Fingertip coordinates (x,y,z)(x, y, z)(x,y,z)
1. Initialize Mediapipe Hands model.
2. Process frame to detect hand landmarks.
3. Extract 21 hand landmarks.
4. Identify fingertip landmarks from the 21 hand landmarks.
5. Normalize coordinates (x,y)(x,y)(x,y) to frame dimensions.
6. Obtain depth coordinate zzz relative to the wrist.

Return: Fingertip coordinates (x,y,z)(x, y, z)(x,y,z).

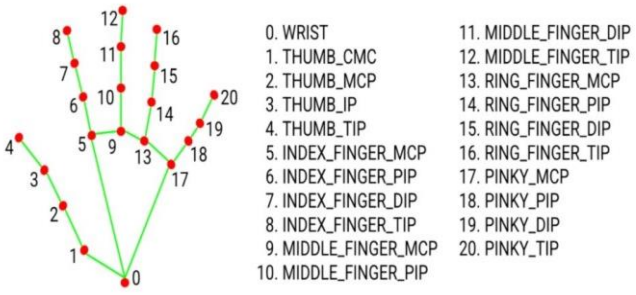

Fig. 1. The 21 specific points on each hand.

B. Fingertip Tracking –

Once the fingertip is successfully detected, the subsequent step is to track its movement to accurately interpret writing gestures. This process begins by continuously monitoring the angle between the joints of the index finger, which is crucial for distinguishing different types of gestures. The system calculates this angle in real-time to track the fingertip's path. To interpret a writing gesture, the system compares the calculated angle with a predefined threshold. If the angle between the joints exceeds this threshold of 160 degrees, it indicates a specific type of movement that the system recognizes as a writing gesture. This threshold is carefully selected to differentiate between various hand movements and gestures. By applying this method, the system can accurately identify writing gestures based on the angle variations, thus enabling precise gesture recognition. The effectiveness of this approach is demonstrated in Fig. 2, which shows how the system responds to different angles and movements.

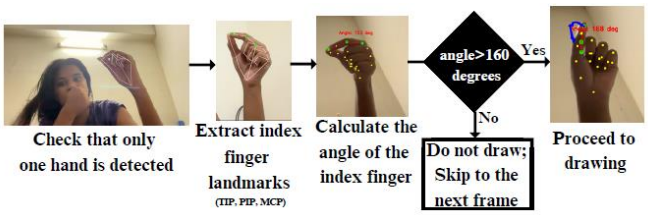

Fig. 2. Finger landmark extraction and angle calculation.

The angle between three points**a, b, c**can be calculated using the law of cosines. The formula is:

$$\theta = \cos^{-1}\left(\frac{|a-b|^2 + |b-c|^2 - |a-c|^2}{2|a-b||b-c|}\right) \quad (1)$$

Where:

$$|a-b| = \sqrt{(a_x - b_x)^2 + (a_y - b_y)^2 + (a_z - b_z)^2} \quad (2)$$

$$|b-c| = \sqrt{(b_x - c_x)^2 + (b_y - c_y)^2 + (b_z - c_z)^2} \quad (3)$$

$$|a-c| = \sqrt{(a_x - c_x)^2 + (a_y - c_y)^2 + (a_z - c_z)^2} \quad (4)$$

Algorithm: Fingertip Tracking and Writing Detection
Input: Video stream from camera
Output: Drawing path
1. Initialize an empty list points to store the drawing path.
2. Initialize last_activity_time.
3. While video capture is open:
a. Read a frame from the camera.
b. Convert the frame to RGB.
c. Process the frame with the Mediapipe Hands model.
d. If hand landmarks are detected:
i. Extract the coordinates of the index finger joints (TIP, PIP, MCP).

International Journal of Engineering Applied Sciences and Technology, 2024
Vol. 9, Issue 04, ISSN No. 2455-2143, Pages 95-99
Published Online August 2024 in IJEAST (http://www.ijeast.com)

ii. Calculate the angle between the joints.
iii. Convert the coordinates to image space.
iv. If the angle is greater than WRITING_ANGLE_THRESHOLD:
- Append the coordinates to the drawing path.
- Update last_activity_time.

v. Else:
- Append None to the drawing path.
e. Display the frame.
f. If 'q' is pressed:
- Break the loop.
Return: Drawing path

C. Text Recognition –
The final step in the process involves recognizing the written text using the Google Cloud Vision API when a pause in writing is detected. An example showing the letter B being air typed is shown in Fig. 3. The system monitors the user's input, and when it detects a significant pause in movement—indicating that writing has likely stopped—it captures the current drawing on the canvas. At this point, the system takes a snapshot of the canvas and saves it as an image. This image, which contains the user's handwriting or drawing, is then sent to the Google Cloud Vision API for text recognition. The API processes the image, analyzing the content to identify and extract any text present. Once the text is recognized, the system can use it for further processing, such as converting handwritten notes into digital text or triggering additional actions based on the recognized content. This automated process ensures that the written input is efficiently converted into machine-readable text with minimal user intervention. The process has been visualized in Fig. 4

Algorithm: Text Recognition
Input: Drawing path
Output: Recognized text
1. Monitor the drawing path for significant pauses in movement.
2. While monitoring the drawing path:
a. If a pause is detected:
i. Capture the current drawing on the canvas.
ii. Save the drawing as an image file.
iii. Send the image file to the Google Cloud Vision API.
iv. Receive and store the recognized text.

Return: Recognized text

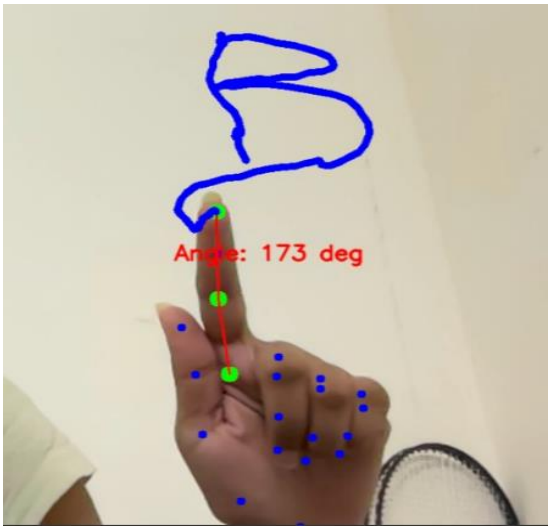

Fig. 3. Air Writing the letter B.

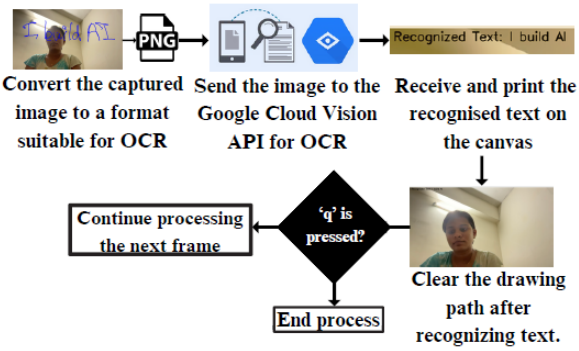

Fig. 4. Flowchart for Text Recognition

Mathematically, the recognition process involves several steps of image processing and pattern matching. These include:

a) **Image Preprocessing:**
The captured image undergoes preprocessing steps to enhance the quality and readability of the text. This includes converting the image to grayscale, binarization using techniques like Otsu's thresholding, and noise reduction using morphological operations. The binary thresholding operation is defined as:

$$I_{binary}(x,y) = 0 \text{ if } (x,y) \leq T \text{ else } 1 \qquad (5)$$

where I(x, y) is the pixel intensity at position (x, y) and T is the threshold value obtained using Otsu's method.

b) **Text Segmantation:**
The preprocessed image is segmented to isolate individual characters or words. This involves identifying connected components and bounding boxes around text regions. The bounding box is defined as:

International Journal of Engineering Applied Sciences and Technology, 2024
Vol. 9, Issue 04, ISSN No. 2455-2143, Pages 95-99
Published Online August 2024 in IJEAST (http://www.ijeast.com)

$$\text{BoundingBox}(x, y, w, h) \quad (6)$$

Where(x, y) are the coordinates of the top-left corner of the bounding box, and (w, h) are the width and height of the box.

c) **Feature Extraction:**
Features are extracted from the segmented text regions to represent the characters. Common features include pixel values, histograms of oriented gradients (HOG), and scale-invariant feature transform (SIFT) descriptors. The Histogram of Oriented Gradients (HOG) is defined as:

$$\text{HOG} = \sum_{i=1}^{n} \left[\left(G_x(i) - G_x(i-1) \right)^2 + \left(G_y(i) - G_y(i-1) \right)^2 \right] \quad (7)$$

d) **Pattern Matching:**
The extracted features are matched against a database of known character features using techniques like k-nearest neighbors (KNN) or neural networks. The matching process involves calculating similarity scores and selecting the best match. The similarity measure is defined as:

$$\text{Similarity}(a, b) = \frac{\sum_{i=1}^{n} a_i^2 \cdot \sum_{i=1}^{n} b_i^2}{\sum_{i=1}^{n} a_i \cdot b_i}$$

where a and bare feature vectors of the characters being compared.

e) **Neural Network Classification:**
In the case of using neural networks, the segmented and preprocessed text image is passed through a series of convolutional layers, pooling layers, and fully connected layers to classify the characters. The equation for the neural network layer is defined as:

$$y = f(Wx + b)$$

where y is the output vector, W is the weight matrix, x is the input vector, b is the bias vector, and f is the activation function (e.g., ReLU, sigmoid).
These mathematical steps and algorithms allow the Google Cloud Vision API to accurately identify and convert handwritten characters into digital text, ensuring the reliable performance of the Visual Air Typing System.

IV.CONCLUSION

In conclusion, the implementation of the Visual Air Typing System highlights the potential of fingertip detection, gesture tracking, and text recognition as a viable alternative to traditional input methods. By leveraging the MediaPipe framework for hand tracking and Google Cloud Vision API for accurate text recognition, the system offers a seamless and intuitive user experience. This technology provides a particularly beneficial solution for individuals with visual impairments, as well as for use in virtual reality environments where traditional keyboards are impractical. The study demonstrates the effectiveness of air typing systems in improving accessibility and user interaction, paving the way for more inclusive and efficient human-computer interaction systems. Future work could focus on further enhancing the system's accuracy and exploring additional applications in diverse technological environments.

V. REFERENCE

[1] Lugaresi, C., Tang, J., Nash, H., McClanahan, C., Uboweja, E., Hays, M., Zhang, F., Chang, C.-L., Yong, M. G., Lee, J., Chang, W.-T., Hua, W., Georg, M., and Grundmann, M. (2019). MediaPipe: A Framework for Building Perception Pipelines, arXiv preprint, arXiv:1906.08172.
[2] https://ai.google.dev/edge/mediapipe/solutions/guide.
[3] https://cloud.google.com/vision?hl=en.
[4] https://microsoft.github.io/data-for-society/dataset?d=MS-ASL-American-Sign-Language-Dataset#overview.
[5] Taskiran, M., Kıllıoğlu, M., and Kahraman, N. (2018). A Real-Time System For Recognition of American Sign Language by Using Deep Learning, in Proc. of the 2018 International Conference on Telecommunications and Signal Processing (TSP).
[6] Sundar, B., and Bagyammal, T. (2022). American Sign Language Recognition for Alphabets Using MediaPipe and LSTM, Procedia Computer Science, 215, (pp. 642-651).
[7] Hochreiter, S., and Schmidhuber, J. (1997). Long Short-term Memory, Neural Computation, 9, (pp. 1735-1780).
[8] Chen, M., AlRegib, G., and Juang, B.-H. (2016). Air-Writing Recognition—Part I: Modeling and Recognition of Characters, Words, and Connecting Motions, IEEE Transactions on Human-Machine Systems, 46(3), (pp. 403-413).
[9] Hsieh, C.-H., Lo, Y.-S., Chen, J.-Y., and Tang, S.-K. (2021). Air-Writing Recognition Based on Deep Convolutional Neural Networks, IEEE Access, 9, (pp. 142827-142836).
[10] Mishra, O., Suryawanshi, P., Singh, Y., and Deokar, S. (2023). A Mediapipe-Based Hand Gesture Recognition Home Automation System, in Proc. of the 2nd International Conference on Futuristic Technologies (INCOFT), Belagavi, Karnataka, India, (pp. 1-6).
[11] Quiñonez, Y., Lizarraga, C., and Aguayo, R. (2022). Machine Learning Solutions with MediaPipe, in Proc. of the 11th International Conference on Software Process Improvement (CIMPS), Acapulco, Guerrero, Mexico, (pp. 212-215).