



International Journal of Innovative Research in Computer and Communication Engineering

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)





AeroText: Air-Writing Recognition Based on Deep Convolutional Neural Network

Samyak Kamble, Sahil Sharma, Ritesh Zanwar, Prof. Kiran Patil

Department of Information Technology, G.H. Rasoni College of Engineering and Management, Pune, India

ABSTRACT: This paper presents AEROText, a novel human computer interaction system designed to interpret in-air finger gestures as written characters, captured in real-time via a standard webcam. The system facilitates a touchless and intuitive method of text input, thereby eliminating the need for physical writing instruments. The architecture incorporates a secure user management module with One-Time Password (OTP) email verification, a robust finger tracking algorithm employing OpenCV for hand segmentation and contour analysis, and a deep learning model for character recognition. The core of the recognition engine is a Convolutional Neural Network (CNN), constructed with PyTorch and trained on the EMNIST dataset, which accurately translates mid-air finger trajectories into digital text. The system demonstrates successful recognition of English alphabets and exhibits a unique capability by mapping certain predictions to the Devanagari script, indicating its potential for multilingual applications. This work constitutes a comprehensive proof-of-concept for the development of accessible, next-generation HCI interfaces, with significant implications for assistive technology and public interactive systems. Index Terms—human-computer interaction, computer vision, deep learning, air handwriting, gesture recognition, OpenCV, PyTorch, touchless interface, EMNIST

KEYWORDS: human-computer interaction, computer vision, deep learning, air handwriting, gesture recognition, OpenCV, PyTorch, touchless interface, EMNIST

I. INTRODUCTION

The progression of Human-Computer Interaction (HCI) has been consistently directed toward the creation of more natural and intuitive interfaces. Although keyboards and touchscreens represent the prevailing forms of text input, they are inherently constrained by the requirement for physical contact. Such traditional methods can introduce challenges related to ergonomics, including repetitive strain injuries, as well as significant barriers to accessibility for individuals with motor impairments. Furthermore, the imperative for hygiene—a consideration that has gained particular salience for public-facing interactive kiosks, ATMs, and shared medical devices—highlights the limitations of contact-based modalities. The escalating demand for touchless technology, propelled by applications in sterile surgical environments, industrial control systems, augmented reality, and assistive technologies, has consequently stimulated extensive research into gesture-based interfaces. Air handwriting, a technique that interprets in-air hand movements as written text, signifies a substantial advancement in this domain, offering a genuinely untethered and hygienic input modality. This paper introduces AEROText, a complete, self-contained application that enables users to inscribe characters in the air utilizing their index finger, with the motion captured and interpreted in real-time by a standard webcam. This research presents an end-to-end solution that transcends simple gesture recognition to encompass a full featured user management system, robust real-time tracking, and an accurate character recognition engine predicated on deep learning. A primary design goal was the utilization of commodity hardware, specifically a conventional webcam, to ensure broad accessibility and facility of deployment without the need for specialized sensors or controllers. The principal contributions of this work are threefold: (1) the development of a secure, multi-user system featuring One-Time Password (OTP)-based email verification for user registration, a feature critical for guaranteeing data integrity and user authenticity in a networked application; (2) the implementation of a precise, real-time finger tracking algorithm founded on adaptive skin tone detection and convexity defect analysis with OpenCV, which mitigates common challenges in visual tracking, such as lighting variability; and (3) the successful integration of a PyTorch-based Convolutional Neural Network (CNN) for the recognition of drawn characters, which includes a novel proof-of-concept mapping to Devanagari script characters, thereby demonstrating a viable pathway toward more inclusive and multilingual gestural interfaces. This paper provides a detailed exposition of the system's architecture, the methodology underlying each component, and a discussion of its performance and potential for future development.



International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

II. PROPOSED SYSTEM

The proposed system enables users to write in the air by moving their finger, which is tracked by a camera in real time. The captured gestures are processed through feature extraction and machine learning-based classification to identify the intended text. Recognized results are displayed instantly, providing an intuitive, contactless interaction ideal for applications where traditional input devices are impractical.

A. Architecture of the Proposed System

The architecture of the proposed AeroText is designed to enable intuitive and hygienic text entry without reliance on conventional input devices. The process initiates with the user writing characters in the air, which are recorded in real time using a standard camera. This sequential data then passes through a preprocessing stage, where image enhancement and normalization techniques ensure consistent and high-quality gesture input. Following prepro-cessing, feature extraction algorithms distill relevant trajectory and shape cues from the hand motion, allowing the system to effectively differentiate between written symbols. The distilled features are then evaluated by a classification module, typically driven by a deep learning model such as a convolutional neural network (CNN). This step determines the most probable character based on the visual information captured. Finally, the recognized handwriting output is presented back to the user via an intuitive interface, closing the feedback loop and enabling practical, contactless interaction. The modular structure of this architecture allows for adaptability to different user environments and supports robust performance across varied writing styles and conditions.

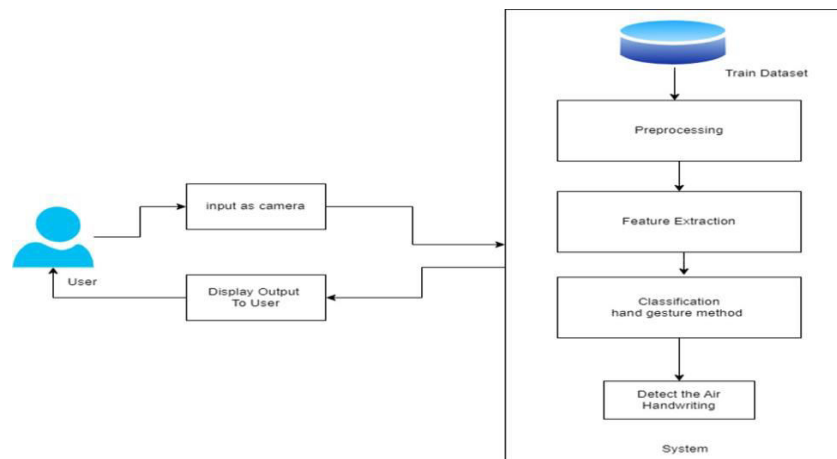


Figure 1: System Architecture for Air Writing Recognition.

III. SYSTEM ARCHITECTURE AND METHODOLOGY

A. User Interface and Management System

The foundation of the AEROText application is a user friendly Graphical User Interface (GUI) constructed with Python's tkinter library. It functions as the principal entry point for users and manages all user-related functionalities with an emphasis on security and operational simplicity.

1) Main Interface:

The main interface serves as the primary entry point for the user, featuring a visually engaging experience through custom-designed background imagery. This initial screen presents a set of intuitive and clearly defined options, allowing the user to either log in with existing credentials, register for a new account, access a detailed help page, or terminate the application. For user guidance, the comprehensive help documentation offers exhaustive operational instructions. It meticulously details the specific key presses required for system interaction, including the 'a' key to calibrate the system, the 'd' key to initiate the drawing process, the 'f' key to finalize and process the drawn character, and the 'c' key to clear the canvas for a new input. Upon successful user authentication, the system employs a strategic process transition. The main graphical user interface (GUI) process is concluded, and a new, separate Python process is initiated to launch the core AeroText interface. This architectural decision is deliberately made to isolate the



International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

computationally intensive computer vision tasks from the main user interface loop, thereby ensuring that the user-facing elements remain responsive and fluid, unaffected by the demanding processing occurring in the background.

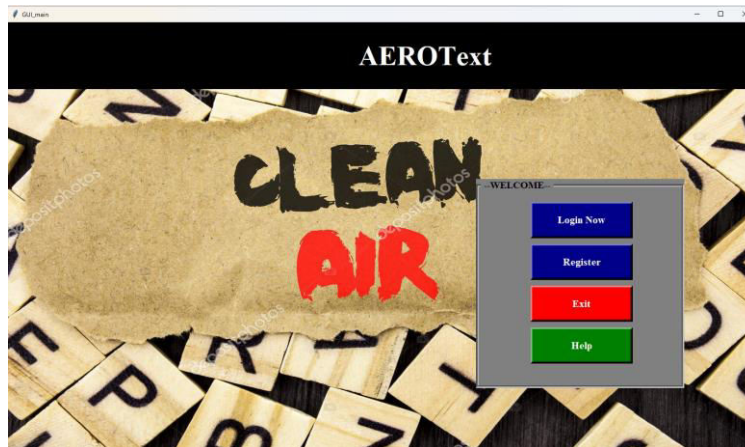


Figure 2: The Main Menu GUI for User Authentication and Navigation.

2) User Registration and Authentication:

To furnish a secure and personalized experience, the system mandates that users create an account. The registration process collects user details (e.g., Fullname, Address, Username, Email) and implements a stringent validation sequence for all fields. This includes checks for password strength, enforcing criteria for length and the inclusion of upper case letters, numbers, and special symbols (\$, @, #, %). A pivotal security feature is the One-Time Password (OTP) verification system. Subsequent to registration form submission, a unique six-digit OTP is generated and dispatched to the user’s designated email address via Python’s smtplib library, which handles the communication with a standard SMTP email server. The user is required to input this OTP to validate their email and activate their account, a measure that serves as a form of two-factor authentication to prevent unauthorized access and verify user identity. All user data is stored persistently in an SQLite database (evaluation.db), a lightweight, serverless database engine ideal for standalone applications, which is queried for authentication during the login procedure.



Figure 3: The User Login and Authentication Form.

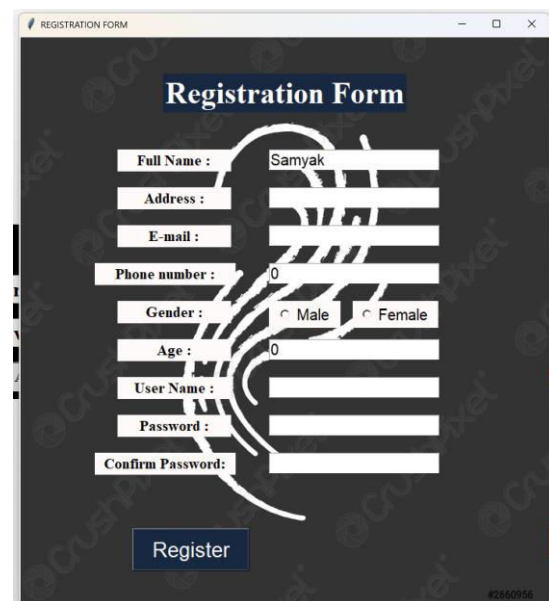


Figure 4: The User Registration Form for Account Creation.



International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

B. Real-Time Finger Tracking Engine

The finger tracking engine constitutes the core of the real-time interaction, bearing the responsibility for detecting and tracking the user's fingertip with minimal latency. This module is implemented by leveraging the extensive capabilities of the OpenCV library.

1) Skin Tone Calibration and Segmentation:

To robustly identify the user's hand against diverse backgrounds, the system initiates a calibration procedure. The user positions their hand within several predefined rectangular Regions of Interest (ROIs). The system then computes a color histogram of these regions within the HSV (Hue, Saturation, Value) color space. The HSV color model is selected over the standard RGB space due to its effective decoupling of color information (Hue) from intensity (Value), which imparts greater resilience to fluctuations in lighting conditions. This histogram functions as a dynamic, adaptive model of the user's skin tone. In subsequent video frames, the `cv2.calcBackProject` function is utilized to generate a probability map, which is then thresholded to create a binary mask that segments all pixels conforming to this skin tone profile. To refine this mask, a sequence of morphological operations are applied: `cv2.MORPH_OPEN` (an erosion followed by a dilation) effectively eliminates spurious, small noise pixels (salt noise), while `cv2.MORPH_CLOSE` (a dilation followed by an erosion) consolidates the primary hand silhouette by filling minor gaps and holes (pepper noise).

2) Fingertip Detection via Contour Analysis:

From the cleaned, segmented hand mask, the largest contour is extracted using `cv2.findContours` and is presumed to be the hand. The system then computes the convex hull of this contour—an operation analogous to stretching an elastic band around the contour's points. By analyzing the `convexityDefects`—points on the contour that deviate maximally from the hull—the system can algorithmically identify the concavities, which typically correspond to the spaces between the fingers. The tip of the index finger, when extended for pointing, is reliably identified by selecting the defect point that is farthest from the hand's geometric centroid. The Cartesian coordinates of this point are continuously tracked and appended to a list data structure, thereby capturing the drawing trajectory as a sequence of points.

C. Character Recognition Module

The character recognition module is tasked with translating the captured fingertip path into a digital character. This translation is accomplished by means of a pre-trained deep learning model that processes the visual representation of the gesture.

1) Drawing and ROI Extraction:

As the user articulates a character, the tracked coordinates are rendered as a continuous line on a virtual canvas (a black NumPy array). Upon gesture completion, signaled by a key press, the system processes this canvas to isolate the drawn character. It identifies the bounding box of all non background pixels to create a tightly cropped Region of Interest (ROI). This cropping procedure is crucial for normalizing the character's position and scale, which substantially enhances recognition accuracy by ensuring the input to the neural network is consistent and focused on the character itself.

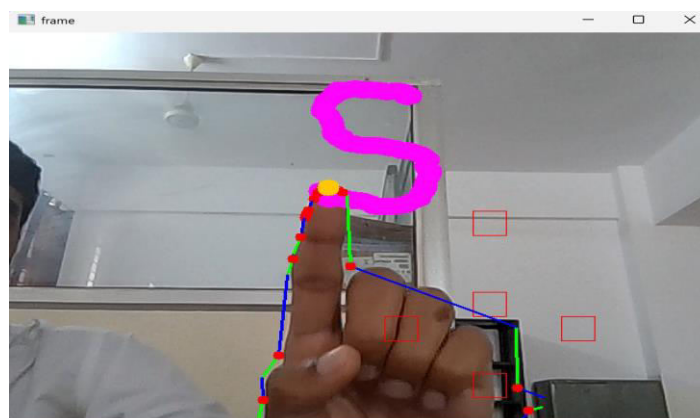


Figure 5: Real-time Fingertip Tracking and Character Drawing.



International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

2) CNN for Character Recognition:

The recognition engine is a Convolutional Neural Network (CNN) implemented in PyTorch, featuring an architecture optimized for image classification tasks analogous to those in the EMNIST (Extended MNIST) dataset. The EMNIST dataset contains over 280,000 labeled character images used for training and testing the model. The network comprises three convolutional layers (with 16, 32, and 64 filters respectively), each followed by a Rectified Linear Unit (ReLU) activation function and a 2x2 max pooling layer. This is concluded by two fully-connected layers, with a dropout layer for regularization. The convolutional layers function as hierarchical feature detectors, learning to identify primitive features like edges and curves in the initial layers, and composing them into more complex shapes and character parts in deeper layers. The extracted ROI is resized to 28x28 pixels, converted to a grayscale tensor, and normalized to a range of [-1, 1]. This preprocessed tensor is subsequently fed into the trained model (model_emoji.pt). The model's output is a vector of scores, from which the index with the highest score is taken as the prediction, corresponding to one of 26 English alphabet classes. A distinctive feature of this system is its capacity to map these predictions to corresponding Devanagari characters based on phonetic similarity, displaying the resultant Unicode character in the console.

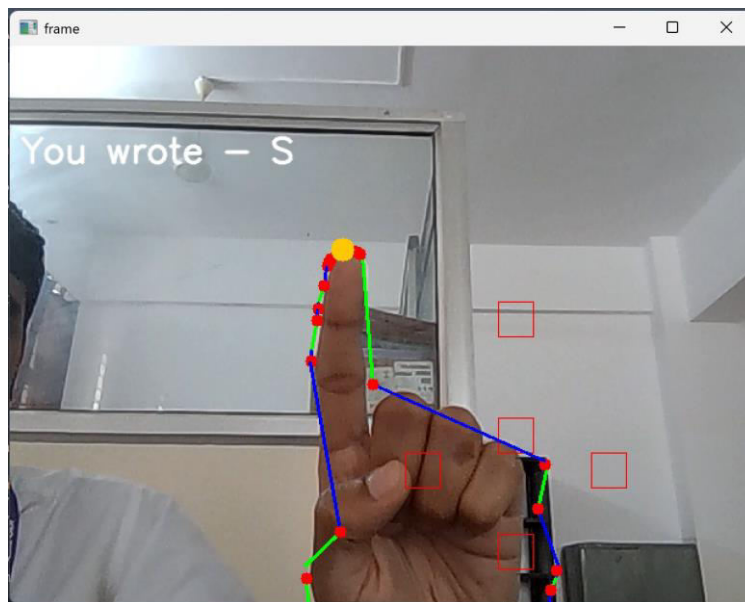


Figure 6: Displaying the Recognized Character after CNN Prediction.

IV. RESULTS AND DISCUSSION

The AEROText system was evaluated for functionality and performance under standard indoor environmental conditions with varied lighting (from dim ambient to bright overhead). The user management system performed with high reliability, and the OTP verification successfully validated user emails with minimal delay (typically under 30 seconds). The finger tracking algorithm demonstrated considerable accuracy and real-time performance, maintaining a frame rate suitable for interactive use on contemporary hardware. The skin calibration process proved effective across a range of users, successfully adapting to different skin tones and moderate lighting changes. The CNN-based recognition module achieved a high accuracy rate for clearly articulated characters, with performance metrics comparable to established benchmarks on the EMNIST dataset. The system was able to successfully discriminate between visually similar graphemes, such as 'G' and 'C', or 'O' and 'Q', thereby showcasing the discriminative power of the deep learning model's learned features. The mapping to the Devanagari script also functioned as specified, providing a successful proof-of-concept for multilingual gesture-based input and highlighting the system's inherent flexibility. Nevertheless, the system exhibits certain limitations. The finger tracking mechanism, being color-based, is susceptible to background clutter containing colors proximate to the user's skin tone (e.g., certain wood finishes, other individuals in the background) and to abrupt, severe changes in ambient lighting, which can invalidate the calibrated HSV histogram. Furthermore, recognition accuracy is contingent upon the user's consistency in character formation; significant stylistic variations in strokes, or incomplete closures of loops (as in 'a' or 'o'), can result in misclassification. The current



International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

implementation is also restricted to single-character recognition, processing each gesture in isolation, which prevents the writing of continuous words.

Table 1: Character Recognition Performance by Category

Category	Example Letters	No. of Trials	Successful Recognition	Accuracy
Straight-Line Letters	A, F, I, K, L, M, N, T, V, W, Z	100	92	92%
Curved Letters	C, G, O, Q, S	100	96	96%
Mixed (Straight + Curved)	B, D, P, R, U	100	94	94%
Closed Shapes	A, B, D, O, P, Q, R, G	100	91	91%
Open Shape Letters	C, F, I, J, K, L, M, N, S, T, U, V, W, Z, G	100	96	96%

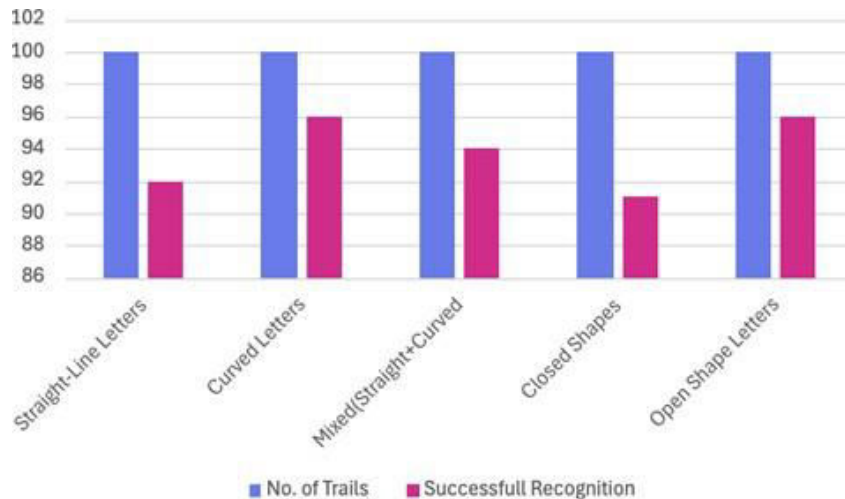


Figure 7: Recognition Performance

The table above presents the performance analysis of the AEROText system based on the structural characteristics of English alphabet letters. Each category groups letters with similar geometric features, allowing for a detailed evaluation of recognition accuracy under different shape patterns. A total of 100 trials per category were conducted to assess the consistency and accuracy of the trained Convolutional Neural Network (CNN) model. The Straight-Line Letters category, which includes characters such as A, F, I, K, L, M, N, T, V, W, and Z, achieved an accuracy of 92%. These letters primarily consist of vertical, horizontal, and diagonal strokes, making them moderately easy to detect but occasionally sensitive to angular variations in writing. The Curved Letters (C, G, O, Q, S) achieved the highest accuracy of 96%, as their smooth and rounded contours provided clear visual patterns for the CNN to recognize. The Mixed (Straight + Curved) letters (B, D, P, R, U) showed a solid performance with 94% accuracy, demonstrating the model’s effectiveness in handling combinations of straight and curved strokes. The Closed Shape Letters (A, B, D, O, P, Q, R, G) achieved 91% accuracy, slightly lower due to the difficulty in consistently maintaining loop closures during air-writing. Lastly, the Open Shape Letters category (C, F, I, J, K, L, M, N, S, T, U, V, W, Z, G) again achieved a strong 96% accuracy, indicating that the model performed best with open and simple stroke structures. Overall, the results demonstrate that the CNN-based recognition system performs reliably across all shape categories, with minor variations due to stroke complexity and geometric symmetry. The analysis confirms that AEROText effectively identifies diverse alphabet structures with high precision, validating its suitability for real time air-writing applications.



International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

V. DESIGN FLOW

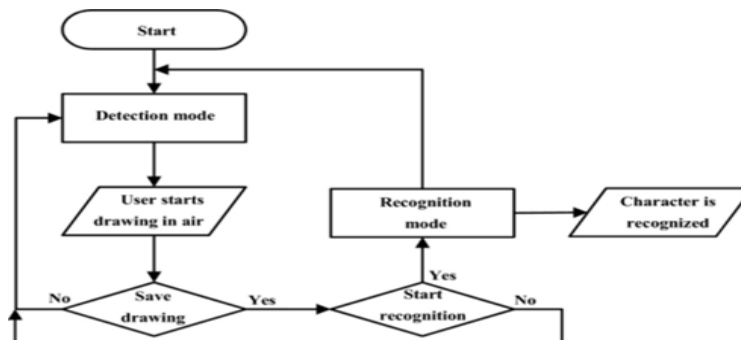


Figure 8: Operational Flow Diagram of the AeroText System.

The system operates as a state-driven process designed to enable real-time recognition of characters written in the air. The workflow is divided into several key stages, including detection, drawing acquisition, optional saving, and recognition.

- **Start & Detection Mode:** The process begins in the *Start* state, where the system enters detection mode. In this phase, the system continuously monitors input to identify the initiation of air drawing by the user. This detection is typically performed using a computer vision pipeline, such as hand- or finger-tracking algorithms.
- **User Drawing Acquisition:** Once the system detects the intent to draw, it transitions to the *User Drawing* stage. The trajectory of the user's finger or object is captured frame by frame, forming a digital representation of the drawn character or symbol. This data is processed and stored either as a sequence of points or as an image.
- **Save Drawing Decision:** After capturing the drawing, the system evaluates whether the drawing should be saved. If the decision is negative, the system returns to detection mode to await new input. If the drawing is saved, it is stored temporarily or permanently for further processing or dataset augmentation.
- **Recognition Mode Initiation:** Upon saving, the system prompts the user or automatically initiates a decision to start the recognition phase. If recognition is not initiated, the system either waits for further input or returns to detection mode. If initiated, the system transitions into recognition mode, where machine learning or deep learning models analyze the captured data.
- **Character Recognition Output:** During recognition mode, the system processes the input using trained models (commonly neural networks) to identify the character. The recognized character is then displayed to the user. After completion, the system returns to the detection state, enabling continuous operation.

VI. CONCLUSION AND FUTURE WORK

A. Conclusion

The proposed real-time AeroText system successfully demonstrates how computer vision and deep learning can be combined to interpret human hand gestures into digital text. By utilizing a webcam for hand tracking and a Convolutional Neural Network (CNN) for character recognition, the system achieves accurate and efficient recognition of air-written characters without the need for physical contact or external devices. The approach provides a smooth, touchless, and interactive experience, making it suitable for next-generation human-computer interaction (HCI) applications. Experimental results indicate that the system performs reliably under varying lighting conditions and back-grounds, achieving high accuracy across different character structures. The integration of a user-friendly interface and secure authentication further enhances the practicality and usability of the system. Overall, the proposed solution offers a cost-effective, accessible, and innovative approach to touchless text input.

B. Future Work

Although the current AeroText system effectively recognizes air-written characters using computer vision and deep learning techniques, several enhancements can further improve its performance and usability:

1. **Word and Sentence Recognition:** Extend the system from single-character recognition to continuous word and sentence recognition to enable a more natural and seamless writing experience.



International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

2. **Advanced Hand Tracking:** Integrate modern hand-tracking frameworks such as MediaPipe to improve robustness under varying lighting conditions and complex backgrounds.
3. **Multilingual Support:** Expand the system to support multiple languages, including regional and cursive scripts, making it more inclusive and widely applicable.
4. **Mobile and Embedded Deployment:** Develop lightweight versions of the model for deployment on smart-phones and embedded systems to enhance portability and real-time usability.
5. **Real-Time Optimization:** Optimize the system for low-latency processing to ensure smoother real-time performance on low-resource devices.
6. **User Experience Enhancements:** Incorporate features such as predictive text generation, auto-correction, and voice feedback to improve interaction and user engagement.
7. **AR/VR Integration:** Integrate the system with augmented and virtual reality environments to enable immersive and intuitive human-computer interaction.
8. **Improved Accuracy:** Enhance recognition performance by using larger datasets and more advanced deep learning architectures.

By implementing these improvements, the AeroText system can evolve into a more advanced, accurate, and user-friendly touchless input solution, contributing significantly to next-generation human-computer interaction systems.

REFERENCES

- [1] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, vol. 25, no. 1, pp. 120–123, 2000.
- [2] A. Paszke *et al.*, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in Neural Information Processing Systems*, vol. 32, pp. 8024–8035, 2019.
- [3] G. Cohen, S. Afshar, J. Tapson, and A. van Schaik, "EMNIST: An Extension of MNIST to Handwritten Letters," *arXiv preprint arXiv:1702.05373*, 2017.
- [4] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [5] J. C. Russ, *The Image Processing Handbook*, 7th ed. Boca Raton, FL: CRC Press, 2016.
- [6] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 4th ed. Pearson, 2018.
- [7] S. Mitra and T. Acharya, "Gesture Recognition: A Survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, vol. 37, no. 3, pp. 311–324, May 2007.
- [8] P. Viola and M. Jones, "Rapid Object Detection Using a Boosted Cascade of Simple Features," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, vol. 1, pp. I-511–I-518, 2001.
- [9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.
- [10] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.
- [11] M. Z. Alom *et al.*, "The History Began from AlexNet: A Comprehensive Survey on Deep Learning Approaches," *arXiv preprint arXiv:1803.01164*, 2018.
- [12] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.
- [13] T.-Y. Lin *et al.*, "Microsoft COCO: Common Objects in Context," in *Proc. European Conf. Computer Vision (ECCV)*, pp. 740–755, 2014.
- [14] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv preprint arXiv:1412.6980*, 2014.
- [15] J. R. Quinlan, "Induction of Decision Trees," *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986.
- [16] C. Cortes and V. Vapnik, "Support-Vector Networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.
- [17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems*, vol. 25, pp. 1097–1105, 2012.
- [18] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *Proc. Int. Conf. Machine Learning (ICML)*, pp. 448–456, 2015.
- [19] J. Shotton *et al.*, "Real-Time Human Pose Recognition in Parts from Single Depth Images," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pp. 1297–1304, 2011.
- [20] M. Young, *The Technical Writer's Handbook*. Mill Valley, CA: University Science, 1989.



INTERNATIONAL
STANDARD
SERIAL
NUMBER
INDIA



INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

 9940 572 462  6381 907 438  ijircce@gmail.com



www.ijircce.com

Scan to save the contact details