IJARSCT

International Journal of Advanced Research in Science, Communication and Technology

International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal

ISSN: 2581-9429

Volume 5, Issue 2, June 2025

Impact Factor: 7.67

Air-Writing Recognition using Deep Learning and Artificial Intelligence

Mohsin Arab, Aman Sande, Sushil Deelip Sonawane, Vikas Mahadev Morabagi, Dr. Vilas Joshi

Department of Computer Engineering

ISBM College of Engineering Nande, Pune, India

mohsinarab063@gmail.com , aamansande6@gmail.com,

sushilsonawane2002@gmail.com,vikasmorabagi24680@gmail.com

Abstract: *This research introduces a real-time air-writing recognition system that allows users to write letters and numbers in the air using intuitive hand gestures captured via a webcam. By fusing computer vision with deep learning techniques, the system bypasses the need for physical contact or writing surfaces, providing a hygienic and accessible solution ideal for interactive learning, accessibility tools, and creative expression. MediaPipe is utilized for efficient hand landmark detection, while OpenCV and PyGame support a dynamic virtual drawing interface. Deep learning models—based on Convolutional Neural Networks (CNNs)—classify air-drawn alphabets and digits with high accuracy using EMNIST and MNIST datasets. A lightweight Flask-based web interface ensures easy deployment across platforms. Our implementation maintains real-time responsiveness, averaging under 100 ms prediction latency while operating smoothly at 30 FPS. Experimental results validate the feasibility and reliability of air-writing recognition in constrained environments, positioning this system as a promising tool for natural human-computer interaction.*

Keywords: *air-writing recognition system*

I. INTRODUCTION

In the age of touchless technology and human-centric computing, air-writing offers a novel, intuitive method of interaction, particularly relevant for scenarios that require hygiene (e.g., post-pandemic environments), accessibility (for people with physical limitations), or mobility (e.g., remote learning). Traditional writing interfaces such as pen and paper or digital styluses are often constrained by the need for contact, limiting their utility in diverse scenarios.

Air-writing leverages computer vision to interpret a user's finger movements in space, simulating handwriting on an invisible canvas. This paper introduces a comprehensive system that implements air-writing recognition using a standard webcam, without relying on specialized hardware like depth sensors or gloves. The core contribution of our work is the seamless integration of gesture tracking with real-time character classification, providing both a drawing and recognition platform.

We emphasize accessibility, cost-effectiveness, and modularity. By employing common libraries—OpenCV for image processing, MediaPipe for gesture detection, TensorFlow/Keras for model inference, and PyGame for interaction—our system can be deployed easily on low-to-mid-range hardware.

The remainder of the paper is organized as follows: Section 2 reviews related work; Section 3 details the system architecture; Section 4 presents the implementation and training setup; Section 5 analyzes the results; and Section 6 outlines applications and future improvements.

II. BACKGROUND AND RELATED WORK

2.1 Air-Writing Interfaces

Several prior works have explored digital handwriting, but many rely on physical interfaces such as touchscreens or styluses. Systems like AirCanvas use basic gesture detection, but lack robust recognition functionality. Other systems incorporate Leap Motion or depth cameras, which, while effective, introduce cost and complexity.

Our work builds on these ideas but distinguishes itself by using only a conventional webcam for input, and incorporating advanced hand tracking and real-time classification.

2.2 Gesture Recognition using MediaPipe

MediaPipe by Google is a powerful cross-platform framework that allows for real-time, high-fidelity hand tracking. It identifies 21 3D hand landmarks, including fingertips and joints. By monitoring the movement and position of key points—particularly the index and middle fingertips—we segment gestures into drawing, selection, and command modes. Its low latency and ease of integration make it well-suited for real-time applications.

2.3 Handwriting Classification with Deep Learning

CNNs have demonstrated state-of-the-art performance in image classification, particularly for handwritten character recognition. We trained two models: one for digit classification (using MNIST) and another for alphabet classification (using EMNIST Balanced). Each model was optimized for real-time inference using lightweight convolutional layers and dropout regularization to maintain both speed and accuracy.

III. SYSTEM ARCHITECTURE

The system consists of five major components:

- **Hand Tracker (MediaPipe)**: Extracts real-time hand landmarks.
- **Gesture Controller**: Determines interaction mode based on hand posture.
- **Drawing Canvas (OpenCV + PyGame)**: Displays user strokes and tools.
- **Recognition Engine (CNN)**: Processes captured image regions for classification.
- **Web Interface (Flask)**: Optional module for deploying the tool as a web app.

Each module interacts seamlessly, with data passing between them in a structured pipeline. Below, we outline each module in detail.

IV. METHODOLOGY

4.1 Hand Detection

The hand tracking system is based on Mediapipe [1], capturing 21 landmarks per hand. Finger status (up/down) is determined to interpret gestures:

- **Two fingers up**: Selection Mode
- **One finger up**: Drawing Mode

4.2 Drawing Mechanism

The finger tip coordinates (index finger) are used as drawing points. For smooth rendering:

- Brush and eraser thickness are defined.
- Gesture switching resets prior drawing points (xp, yp) for continuity.

4.3 Mode Switching

Users can switch between:

- Alphabet recognition: Press 'A'
- Number recognition: Press 'N'
- Turn-off prediction: Press 'O'

These modes control whether the drawn area will be sent to the prediction model.

4.4 Prediction Models

Two separate CNN models are used:

Alphabet Recognition Model (bModel.h5)

ISSN
2581-9429
IJARSCT

Digit Recognition Model (bestmodel.h5)

Training Details

- Input: Grayscale images (28×28)
- Architecture: Convolutional Neural Networks with ReLU and SoftMax layers.
- Datasets: Modified EMNIST [5] and MNIST [6]

V. IMPLEMENTATION DETAILS

5.1 Environment Setup

We used Python 3.9, with essential libraries including:

- OpenCV for frame capture and processing
- MediaPipe for hand landmark detection
- TensorFlow/Keras for deep learning
- NumPy for data manipulation
- PyGame for UI rendering
- Flask for optional web integration
- Hardware used:
- Laptop with Intel i5 CPU, 8GB RAM
- 720p webcam (built-in)

5.2 Canvas Initialization and Tools

The virtual canvas, built using OpenCV, has a resolution of 1280x720 pixels. Users interact using hand gestures:

- One finger (index) for drawing
- Two fingers (index + middle) for mode selection
- Special areas (top-left) for color and tool selection

Drawn strokes are recorded and rendered in real-time. A separate region is monitored for triggering recognition, allowing the system to extract and preprocess drawn characters

VI. RECOGNITION MODELS AND TRAINING

We trained two separate Convolutional Neural Network (CNN) models to classify hand-drawn characters captured through air-writing.

6.1 Digit Recognition Model

Dataset: MNIST (Modified National Institute of Standards and Technology database) containing 60,000 training and 10,000 test images of digits (0–9).

Architecture:
Input layer: 28×28 grayscale image
Two convolutional layers (32 and 64 filters, kernel size 3×3)
MaxPooling layer
Flatten layer
Fully connected layers (128 units → 10 output classes with softmax)

Training:
Optimizer: Adam
Epochs: 10
Accuracy: ~99% on validation data

6.2 Alphabet Recognition Model

Dataset: EMNIST (Extended MNIST) Balanced subset, including 47 classes (A–Z + lowercase variants).
Architecture: Similar to digit model but modified output layer to match 47 classes.
Training Accuracy: ~95%
Models were saved as .h5 files and loaded during runtime for real-time prediction. Image preprocessing includes padding, grayscale conversion, and normalization before feeding into the models.

VII. DATA PIPELINE AND INTERACTION LOGIC

The system runs a continuous loop with the following pipeline:
Frame Capture: Video stream from webcam at 30 FPS using OpenCV.
Hand Detection: MediaPipe identifies key landmarks in each frame.
Gesture Interpretation:

- One raised finger → draw mode
- Two raised fingers → selection mode
- Three fingers → trigger recognition

Stroke Collection: Coordinates are stored in a list and rendered on the canvas.
Bounding Box Extraction: When recognition is invoked, the system computes the bounding rectangle of the stroke.
Image Preprocessing:

- Resizes ROI to 28×28
- Applies binary threshold
- Applies padding for clarity

Model Prediction:

- ROI is passed to the respective model (digit or alphabet) depending on mode
- Prediction is displayed with a bounding box and label

Reset Option: Users can clear canvas or switch tools through gesture-based menu at the top of the screen.

VIII. RESULTS AND PERFORMANCE EVALUATION

8.1 Testing Environment

- Intel Core i5 CPU, 8GB RAM
- Integrated 720p webcam
- Python 3.9 virtual environment

8.2 Evaluation Metrics

- **Frame rate**: Maintained at ~30 FPS in draw mode
- **Prediction latency**: <100ms

Average accuracy:
Digits: 98.9%
Alphabets: 93.2% (higher confusion for similar-shaped letters like O and Q, M and N)

8.3 Qualitative Observations

- Smooth drawing experience even with fast finger movement
- Reliable gesture detection under varied lighting
- Occasional false positives when hand partially visible — resolved with improved bounding box filtering

8.4 Usability Feedback

- Tested by 10 users (ages 12–40)
- 90% found interaction intuitive

- 85% found predictions accurate enough for casual usage

IX. APPLICATIONS, LIMITATIONS, AND FUTURE WORK

9.1 Applications

- **Education**: Interactive learning tools for children to practice letters and digits
- **Accessibility**: Alternative input for users with motor disabilities
- **Art & Creativity**: Touchless sketching and digital graffiti tools
- **Remote Learning**: Air-writing via webcam for live annotation

9.2 Limitations

- Susceptible to lighting variation and background noise
- Similar characters may confuse CNN due to poor stroke distinction
- Hand occlusion or overlap can impact landmark detection

9.3 Future Improvements

- **Continuous Word Recognition**: Add LSTM or Transformer models for sequential recognition
- **Voice Output**: Enable audio feedback using TTS (text-to-speech)
- **Multi-language Support**: Extend models to Devanagari, Arabic, etc.
- **Mobile/AR Integration**: Deploy to mobile devices or AR headsets for immersive applications

X. CONCLUSION

We presented a novel air-writing recognition system that combines real-time hand gesture tracking with deep learning-based character recognition. It enables intuitive interaction using only a standard webcam, without relying on physical tools or touch surfaces. The modular design allows for extensibility into various domains, including education, accessibility, and creative arts. Future enhancements could position this system as a robust platform for touchless human-computer interaction.

REFERENCES

[1]. Lugaresi, C., et al. (2019). "Media pipe: A Framework for Building Perception Pipelines." arXiv:1906.08172.
[2]. Bradski, G. (2000). "The OpenCV Library." Dr. Dobb's Journal of Software Tools.
[3]. Paul, S., & Sharma, V. (2018). “Air Draw: Hand Gesture Recognition for Drawing and Digit Classification.”IJCA.
[4]. Zhang, X., et al. (2017). "Hand Track: Real-Time Hand Gesture Recognition using Deep Learning." IEEEAccess.
[5]. Cohen, G., et al. (2017). “EMNIST: Extended MNIST Dataset.” arXiv:1702.05373.
[6]. LeCun, Y., et al. (1998). “Gradient-Based Learning Applied to Document Recognition.” Proceedings of the IEEE.
[7]. Ohn-Bar, E., & Trivedi, M. M. (2014). "Hand gesture recognition in real time for automotive interfaces: A multimodal vision-based approach and evaluations." *IEEE Transactions on Intelligent Transportation Systems*, 15(6), 2368–2377. https://doi.org/10.1109/TITS.2014.2320139
[8]. Simonyan, K., & Zisserman, A. (2015). "Very Deep Convolutional Networks for Large-Scale Image Recognition." *International Conference on Learning Representations (ICLR)*.
[9]. Starner, T., Weaver, J., & Pentland, A. (1998). "Real-Time American Sign Language Recognition Using Desk and Wearable Computer Based Video." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12), 1371–1375. https://doi.org/10.1109/34.735811

[10]. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). "ImageNet classification with deep convolutional neural networks." *Advances in Neural Information Processing Systems*, 25. https://doi.org/10.1145/3065386

[11]. Molchanov, P., Gupta, S., Kim, K., & Kautz, J. (2015). "Hand gesture recognition with 3D convolutional neural networks." *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 1–7. https://doi.org/10.1109/CVPRW.2015.7301347

[12]. Kim, J., & Kim, H. (2019). "Hand Gesture Recognition Using Finger Coordinates." *Sensors*, 19(18), 4057. https://doi.org/10.3390/s19184057

[13]. Graves, A., & Schmid Huber, J. (2009). "Offline Handwriting Recognition with Multidimensional Recurrent Neural Networks." *Neural Information Processing Systems (NeurIPS)*.

ISSN 2581-9429 IJARSCT