IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Air-Writing And Recognition System

[1] Dhakshith N, [2] Lavanya A

[1]Student, [2]Assistant Professor

[1]Department Of Computer Science and Engineering, [1,2]Adhiyamaan

College Of Engineering, Hosur, Tamil Nadu

Abstract: The trajectory-based writing system offers a novel approach to writing linguistic characters or words in open space by utilizing finger, marker, or handheld device movements. It addresses challenges faced by traditional pen-up and pen-down writing methods and presents distinct advantages over gesture-based systems due to its simplicity. However, its efficacy is hampered by the complexity of non-uniform characters and varied writing styles. This study presents the development of an air-writing recognition system utilizing three-dimensional (3D) trajectories captured by a depth camera monitoring fingertip movements. To enhance feature selection, we utilized nearest neighbor and root point translation techniques for trajectory normalization. Our recognition model integrates long short-term memory (LSTM) and convolutional neural network (CNN) architectures. Validation was conducted using a self-collected dataset, and robustness was assessed using the 6D motion gesture (6DMG) alphanumeric character dataset, achieving a remarkable 99.32% accuracy, a record high. This underscores the model's robustness across digits and characters. Additionally, we release a dataset comprising 21,000 digits, addressing the current research's dataset scarcity.

***Index Terms* -** air-writing; digit identification; depth sensing; Time-of-Flight (TOF) camera; LSTM; CNN; trajectory analysis; human-machine

I. INTRODUCTION

Air-writing, a method of writing in three-dimensional space through gestures or trajectory information, presents a touchless alternative to traditional writing systems. Particularly valuable in scenarios where conventional methods face limitations such as gesture-based interaction, augmented reality (AR), and virtual reality (VR), air-writing addresses these challenges effectively. Unlike pen-and-paper systems that allow multi-stroke character formation, air-writing lacks this flexibility, which stands as its primary drawback. While gesture-based writing offers a solution, its effectiveness is hindered by limited gesture options, often constrained by human posture. Combining gestures can expand the repertoire, albeit at the cost of increased complexity, posing challenges for new users to grasp.

In contrast, air-writing allows users to follow familiar sequential writing patterns, akin to traditional methods. The trajectory-based approach typically involves writing within an imaginary spatial boundary, whether in front of a 2D or 3D camera. Challenges arise from spatial and temporal variability, as users exhibit diverse writing styles and speeds, resulting in varied trajectories. Recent advancements in 3D vision technology, facilitated by devices like Leap Motion, Kinect, and Intel RealSense cameras, have revolutionized trajectory tracking. While gloves, wearables, and Wi-Fi-based sensors offer accurate trajectory data, they often lack user-friendliness. Leap Motion excels in precise finger tracking, Kinect in long-range tracking, and Intel RealSense strikes a balance between accuracy and range, catering to diverse user needs.

Despite the complexity of trajectory estimation, modern sensors provide reliable trajectory data, albeit with occasional zigzag effects due to unique writing styles. The absence of fixed boundaries in the imaginary writing space contributes to shaky trajectories and unwanted sequences, necessitating normalization techniques to enhance accuracy. Machine learning-based approaches demand meticulous feature selection, a challenging task that deep learning models mitigate by automatically learning relevant features. In this study, we implement two efficient deep learning algorithms, namely the long short-term memory recurrent neural network (LSTM) and the convolutional neural network (CNN), where LSTM demonstrates superior performance.

The contributions of this paper are threefold: (1) Designing two robust deep learning models, LSTM and CNN, for accurate air-writing digit recognition, showcasing resilience across various experimental conditions, especially in normalized trajectory scenarios; (2) Creating a large publicly available dataset comprising 21,000 trajectories, essential for training deep learning algorithms, with an additional 1000 test trajectories to evaluate model performance on unseen data; (3) Validating the accuracy and feasibility of the proposed model using both our dataset and the publicly available 6DMG dataset, demonstrating superior performance compared to prior work.

The remainder of this paper is structured as follows: Section 2 discusses related prior work; Section 3 details the methodology, encompassing trajectory collection and network design; Sections 4 and 5 present experimental procedures and results, respectively, including performance comparisons with publicly available datasets. Finally, Section 6 concludes the paper and outlines avenues for future research.

II. RELATED WORK

Various standalone devices have been explored for air-writing and gesture-based digit recognition, catering to different needs and budgets. These devices, ranging from MEMS gyroscopes to Wi-fi-based systems, offer diverse options for trajectory estimation in three-dimensional space. In particular, Wi-fi-based systems, such as the Wri-Fi system, utilize channel state information derived from wireless signals for device-free air-writing experiences. Portable devices equipped with inertial sensors have gained traction for collecting human motion trajectory data, focusing on minimizing errors through signal manipulation and algorithm refinement. However, challenges persist, including user discomfort with wearable sensors and complexities in reducing trajectory errors.

Methods like gyroscope-based and accelerometer-based approaches leverage extended Kalman filters and wireless transmitters for motion detection, facilitating applications in digital pens and hand gesture recognition. Commercial devices like the Wiimote and HP Slate offer motion detection capabilities, with studies showcasing the effectiveness of the six-degree-of-freedom (6DMG) dataset collected by the Wiimote. Despite promising results, handheld sensors pose usability challenges, leading to the exploration of 3D vision-based approaches using depth sensor-based cameras like the time-of-flight (TOF) camera, renowned for its accuracy in low-light conditions.

Several algorithms have been employed for air-writing recognition, including hidden Markov models (HMM), dynamic time warping (DTW), support vector machines (SVM), and bi-directional LSTM (BLSTM). While HMM and SVM-based approaches are widely used, LSTM and CNN algorithms have gained popularity for their efficacy in various applications, including air-writing recognition. Recent studies have explored novel strategies like shape writing and IMU-based systems, although challenges such as device attachment and recognition accuracy persist.

Efforts have been made to overcome these challenges, with researchers proposing Kinect-based systems for online digit recognition and Persian number recognition. Full writing systems in the air have also been developed, employing trajectory detection devices like Leap Motion and recognition algorithms such as HMM and BLSTM. While these systems show promise, achieving significant recognition accuracy remains a continual endeavor in the field of air-writing recognition.

III. OBJECTIVE

The objective of this study is to develop and validate an air-writing recognition system utilizing three-dimensional trajectories captured by a depth camera. The aim is to address the limitations of traditional writing methods and gesture-based systems by presenting a novel approach that offers simplicity and high accuracy in recognizing linguistic characters or words written in open space. The study focuses on enhancing feature selection, utilizing trajectory normalization techniques, and integrating deep learning architectures (LSTM and CNN) for robust recognition across varied writing styles and non-uniform characters. Validation is conducted using self-collected datasets and the 6D motion gesture alphanumeric character dataset, with the primary goal of achieving a remarkable accuracy rate of 99.32% and releasing a dataset to mitigate dataset scarcity in this area of research.

IV. RESEARCH METHODOLOGY

The entire procedure consists of four key components: fingertip identification, data gathering, standardization, and network architecture. A comprehensive schematic is depicted in Figure 1 to elucidate the specifics of the proposed approach. Subsequent sections offer detailed insights into each component.

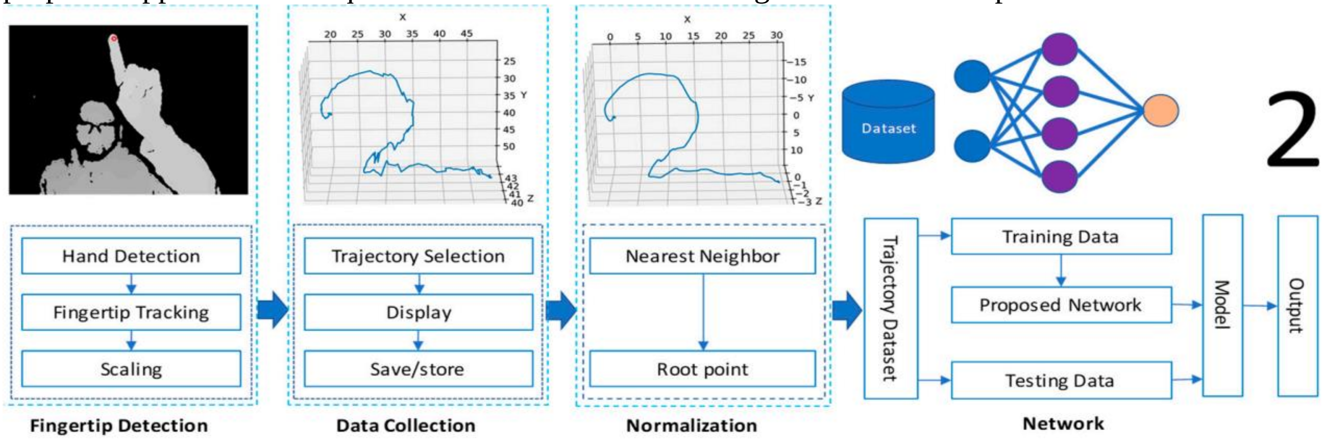


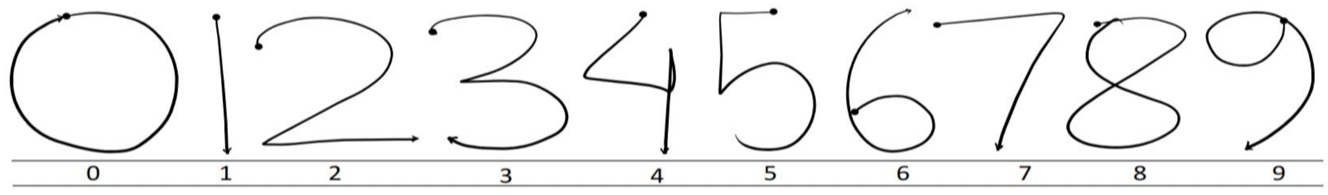
Figure 1: Block diagram for the proposed method including the fifingertip detection, data collection, normalization, and network design.

4.1. Fingertip Identification

For fingertip detection, we utilized the Intel RealSense SR300 camera, renowned for its application in gesture recognition, finger joint tracking, and depth-sensing research. Initially, the process involved hand segmentation and detection. Typically, human hands comprise 22 finger joints, with focus placed on tracking the index fingertip for trajectory writing, ensuring user convenience. However, the trajectory was mapped onto a virtual window, necessitating adjustments to fit and display it within a physical window, such as a computer screen. To achieve this, scaling was implemented. Physical distances were calculated by multiplying the window size by an adaptive value determined for both the horizontal (x) and vertical (y) directions. This adaptive value, ranging between 0 and 1, was obtained from data collected via the RealSense camera. In the User Interface (UI), window sizes of 640 and 480 were designated for the x and y axes, respectively. Original pixel values were computed by multiplying the window size by the adaptive value.

4.2 Data Acquisition

We devised a straightforward interactive user interface (UI) requiring minimal instruction. Ten participants (8 males and 2 females), aged between 23 and 30, all graduate students, were enlisted for the study. The prescribed writing order, depicted in Figure 2, closely resembled traditional writing conventions, without imposing constraints like graffiti or uni-stroke techniques. Stroke orders for each letter adhered to the conventional box-writing method. In recognizing air-writing systems, two main approaches exist: online and offline. For our study, we adopted the offline method, also referred to as the 'push-to-write' approach, to gather data. In this procedure, participants were instructed to write digits in front of the depth camera, capturing their spatial trajectory sequences.



The dataset parameters for the collected RealSense trajectory digits (RTD) are outlined in Table 1. Notably, digits 4, 5, and 8 exhibit relatively longer trajectories compared to others, with digit 1 displaying the shortest trajectory and digit 4 the longest. Users had the flexibility to utilize varying writing areas according to personal preference, resulting in a wide range of trajectory lengths for the same digit. Despite this variability, the mean (Equation 1) and standard deviation (STD) suggest that the majority of data points fall within a reasonable range. The dataset's variance (Equation 2) indicates the extent of data spread, crucial for designing the input layer of deep learning algorithms. The STD (Equation 3) represents deviation from the mean value, offering insights into data distribution characteristics essential for algorithm development.

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$\sigma^2 = \frac{1}{n-1}\sum_{i=1}^{n} (x_i - \bar{x})^2$$

$$\sigma = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n} (x_i - \bar{x})^2}$$

where x is the mean value, xi is the ith datum, σ2 is the variance, and σ is the STD. Some data were collected for testing purposes to verify the accuracy of unknown data.

Digit	Maximum	Minimum	Mean	Variance	STD
0	224	26	60	619	24
1	144	13	36	232	15
2	228	26	73	676	26
3	218	35	74	614	24
4	255	39	90	729	27
5	265	37	80	613	24
6	235	25	58	414	20
7	155	21	53	314	17
8	250	30	80	589	24
9	223	22	64	482	21

4.3 Normalization

A primary challenge encountered in air-writing is the inherent zigzag nature of trajectories, lacking smoothness, necessitating normalization prior to inputting into the network. We employed two normalization methodologies—the nearest neighbor and root point. The procedures are detailed as follows:

4.3.2. Baseline Point Normalization

In air-writing, users employ an imaginary ("virtual") box in which they write their digits. While all digits are written in the first quadrant of a Cartesian plane, their positions may vary within this virtual box, as it lacks fixed boundaries or margins. Consequently, the same digit may be written in different positions, even by the same user, resulting in random initial positions. To address this variability, we utilized baseline point translation to standardize the starting point. This ensures that all trajectories commence from a consistent root coordinate. Equations (7) to (9) are utilized to compute the baseline point.

Here, [x0, y0, z0] denotes the initial point of a sequence, while [xi, yi, zi] represents the instantaneous point derived from the entire trajectory. The unnormalized digit 0 is depicted in Figure 3a, exhibiting noise and zigzag effects, whereas the fully standardized trajectory (Figure 3b) appears smooth. In this representation, 'x' and 'y' denote the distance values within the virtual window, representing the distance between the start and end positions. 'z' represents the distance between the hand fingertip and the camera, all measured in centimeters (cm). Negative values in Figure 3b indicate relative distances, a result of employing Equations (7) to (9) during the normalization process. The geometric baseline point serves as the reference starting point; for instance, -5 in the x-direction, -30 in the y-direction, and -6 in the z-direction signify deviations of 5 cm left, 30 cm below, and 6 cm closer to the camera from the starting position, respectively. Here, the negative sign indicates direction, not mathematical operation.

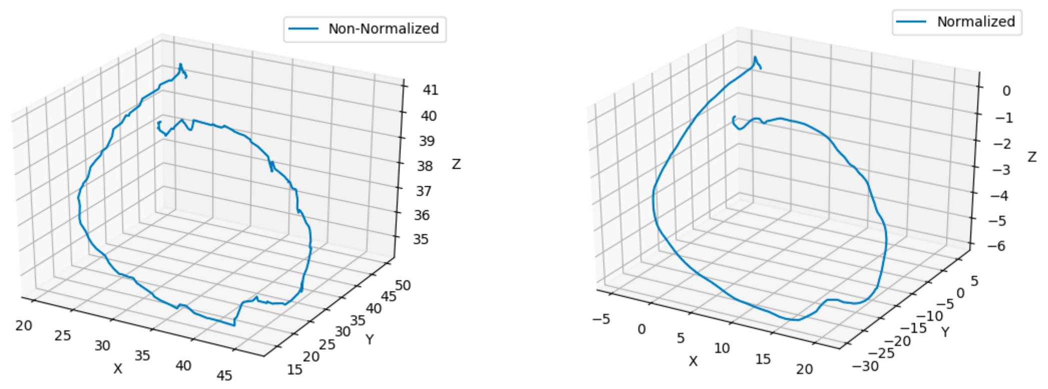


Figure 3. Three-dimensional trajectory visualization: (a) sample trajectory before normalization and (b) trajectory after normalization.

4.4. The Data Collection

In our research, we utilize two distinct datasets: one is our self-compiled collection (RTD), and the other is the 6DMG dataset [33]. Below are the detailed descriptions of each dataset:

4.4.1. RTD Dataset

The RTD dataset comprises trajectories gathered in-house, encompassing 20,000 trajectories, with 2000 data points allocated for each digit. The writing sequence and specific parameters are depicted in Figure 2 and Table 1, respectively. To the best of our knowledge, this dataset boasts the largest digit repository currently available, considering the number of trajectories attributed to each digit. The RTD dataset is readily accessible online, along with comprehensive usage instructions.

4.4.2. 6DMG Dataset

The 6DMG dataset consists of motion gesture data, including alphanumeric air-writing samples [33]. Collected via the Wiimote device by 28 participants, this handheld gadget operates based on acceleration and angular speed, with orientation computed using inertial measurements. For our study, we incorporate the 6DMG air-writing dataset, which features characters written in a uni-stroke fashion. Sample characters are illustrated in Figure 4.

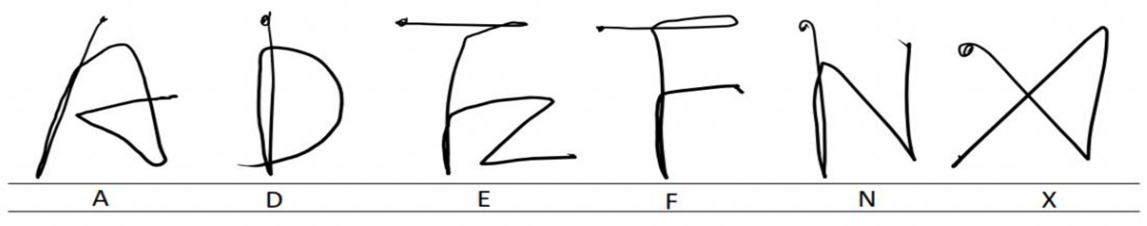


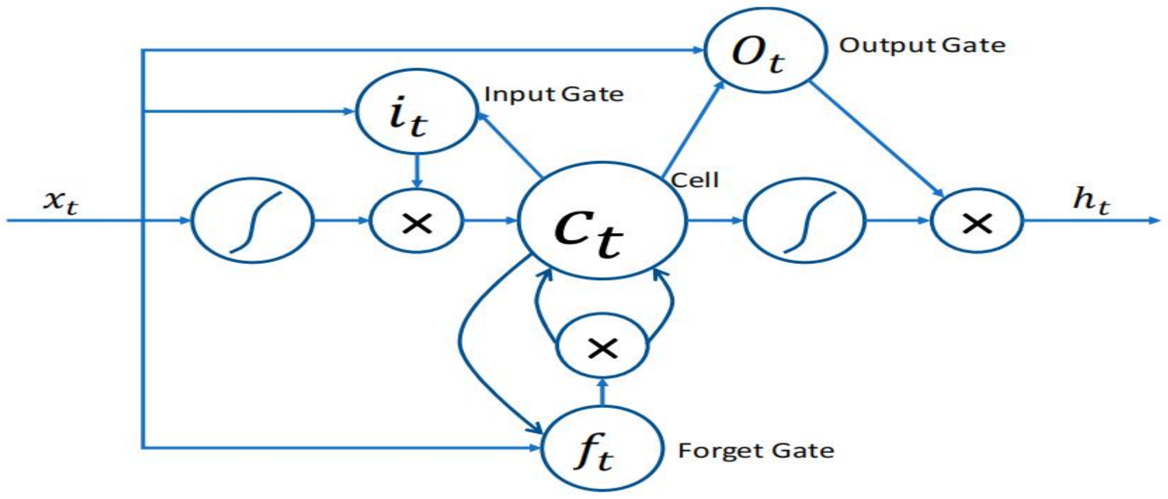
Figure 4. Sample character from 6DMG dataset

4.5. Network Architecture

In this study, we utilized two cutting-edge neural network algorithms: CNN and LSTM. These algorithms leverage convolution and recurrent units, respectively, and are renowned for their efficacy in time series prediction tasks.

4.5.1. LSTM Model

The LSTM (Long Short-Term Memory) network [34] represents a variation of recurrent neural networks (RNNs). Diverging from conventional feed-forward networks, LSTM incorporates feedback connections, rendering it highly significant. Figure 5 illustrates the architecture of an LSTM cell



.

Figure 5. A Single long short-term memory (LSTM) unit. Here, ct, it, ft, and ot represents cell, input gate, forget gate, and output gate, respectively

The LSTM layer consists of a cell housing input (Equation 10), forget (Equation 11), and output (Equation 12) gates, collectively referred to as the cell state (Equation 13). This state undergoes modifications via the forget gate, which determines what information to discard from the previous state, thereby retaining only relevant data. Subsequently, the
output gate of the cell facilitates the transfer of information to the subsequent state.

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i)$$

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f)$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o)$$

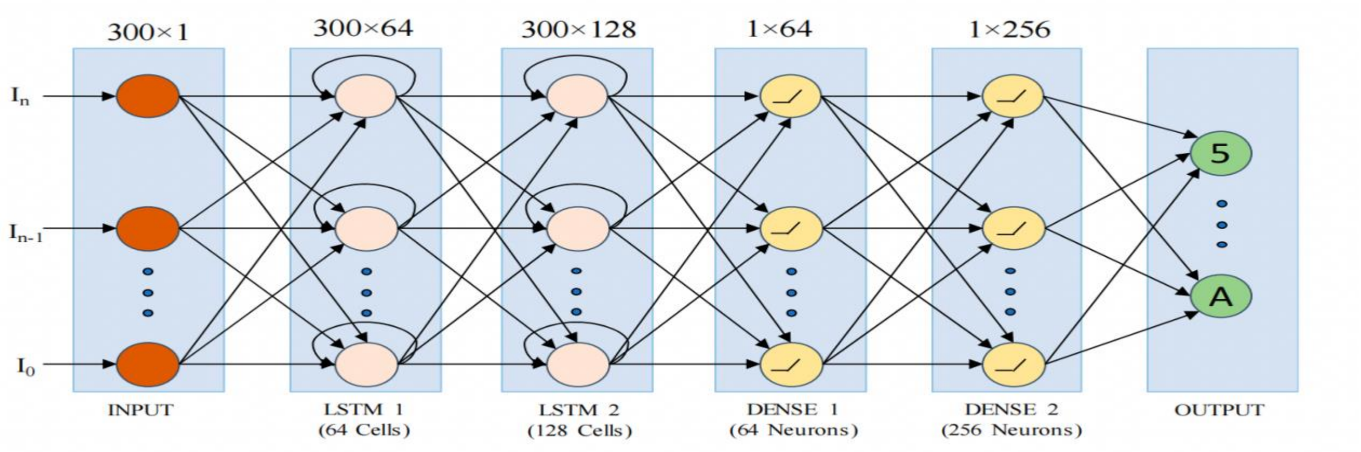
$$c_t = f_t \circ c_{t-1} + i_t \odot \sigma(W_c x_t + U_c h_{t-1} + b_c)$$

where i, f, and o represent the input, forget, and output gates, respectively. The weight matrices W and bias b are parameters that undergo learning during training. The symbol $\odot$ denotes the Hadamard product [35], generating a matrix of the same dimensions as the input.

The proposed network comprises two LSTM layers and two dense layers. The input layer, positioned at the beginning of the network, serves solely to transfer inputs to the subsequent LSTM layer. Given the varying trajectory lengths, each trajectory contains a unique set of 3D spatial points, and even identical digits may exhibit differing lengths based on writing speed and direction. However, LSTM-RNNs necessitate fixed-length sequences as input. Hence, the input layer is configured to accommodate the maximum trajectory length.

The architecture of the proposed LSTM model is depicted in Figure 6; where I0 and In denote inputs containing the x, y, and z values derived from the trajectory. LSTM 1 and LSTM 2 comprise 64 and 128 cells, respectively, with each cell computing the output activation of the LSTM unit. Cross-validation was facilitated using a mini-batch size of 256. DENSE 1 and DENSE 2 house 64 and 256 neurons, respectively, employing the Rectified Linear Unit (ReLU) activation function (Equation 14). ReLU is chosen for its simplicity and non-saturating output characteristic, transmitting only the positive output to the subsequent layer.

Dropout serves as a regularization technique, originating from Google, to mitigate overfitting in neural networks by discouraging complex co-adaptations in training data. It aids in preventing overfitting [36] with a mini-batch size of 256. DENSE 1 and DENSE 2 consist of 64 and 256 neurons, respectively, both employing the Rectified Linear Unit (ReLU) activation function (Equation 14), which is straightforward and non-saturating. The key property of ReLU is its ability to eliminate the negative output portion, transmitting only the positive values to the subsequent layer



.

Figure 6. Network diagram of the proposed LSTM network indicating the input, output and hidden layers. Here I0 and In is the fifirst and last point for an individual trajectory, respectively. There are two LSTM and dense layer with multiple neurons. The output layer is the result of the model.

4.5.2. CNN Architecture

A CNN architecture comprises input and output layers alongside multiple hidden layers, incorporating convolution, pooling, and activation layers. CNNs exhibit various variants tailored to specific applications and datasets. The conventional CNN network entails significant computational overhead due to convolution and pooling layers. Therefore, we employed the separable convolution layer, also known as depth-wise convolution, recognized for its faster computation.

The proposed CNN network is delineated in Figure 7. Similar to the LSTM network, the input layer is configured to accommodate the maximum trajectory length. Comprising a 300 × 1-dimensional vector, the input layer transfers input values to the first separable convolution layer. Convolution layers 1, 2, and 3 feature 64, 128, and 256 channels, respectively, with a filter size of 3 across all cases. Each convolution layer is paired with a 1D max-pooling layer. Two dense layers housing 256 and 128 neurons, respectively, are included. Dropout rate of 0.5 is implemented in DENSE1 and DENSE2 layers. The ReLU activation function (Equation 14) is applied across all layers except the output layer, where softmax (Equation 15) regularization is utilized. The Adam optimizer with a learning rate of 0.0001 and categorical cross-entropy as the loss function are employed.

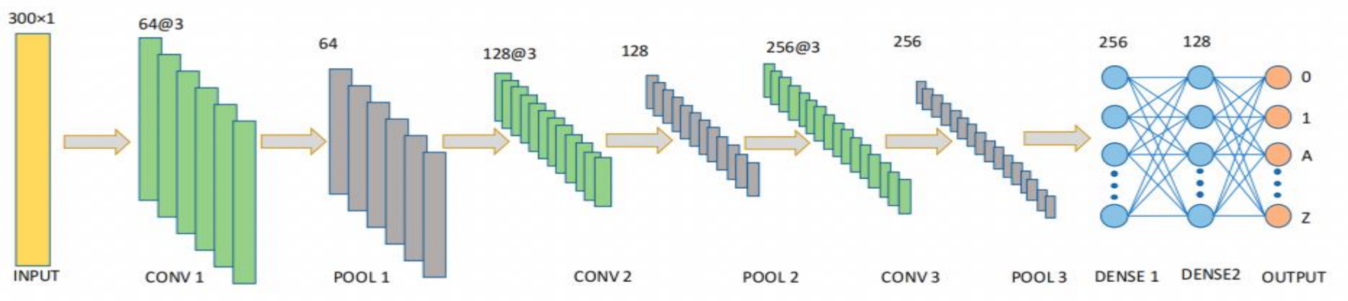


Figure 7. Proposed convolutional neural network (CNN) model. The model has three convolutions,

three max-pooling and 2 dense layers.

V. Experimental Configuration

For the experiment, we connected an Intel RealSense SR300 camera to a computer. A user-friendly graphical interface (GUI) was developed to facilitate trajectory data collection. C# and Python programming languages were utilized for interface development and training, respectively. The proposed networks were implemented using the Keras high-level API with TensorFlow backend. Real-time trajectory capture was achieved at a rate of 50 frames per second (fps). To expedite the training process, we employed an NVIDIA GeForce GTX 1050 Ti graphics processing unit (GPU) with 32 GB memory. The experimental setup is illustrated in Figure 8.

The trajectory capturing process commenced with writing the digit 0 in the air, tracked by the RealSense camera. Trajectories were captured in real-time without normalization, resulting in slight distortions. Complex digits were also captured frame by frame, aiding in understanding the motion associated with each digit's writing order.

5.1. Interface Design

The user interface featured three primary buttons: start, stop, and save trajectory. It displayed both depth information and the captured trajectory, offering real-time trajectory visualization. The GUI was intuitive and interactive, requiring minimal instruction. Each digit was represented in 2D Cartesian coordinates for simplicity, though the data was inherently three-dimensional. Upon clicking the 'start' button, the camera initiated fifingertip position detection. Users could stop trajectory recording by clicking the 'stop' button, refresh the window if needed, and save trajectories using the 'save' button. Labels for the trajectories were automatically generated by the system. The 'start' and 'stop' buttons controlled the initial and end points, respectively, while the 'cut' button facilitated trimming of unwanted starting or ending points.

5.2. User Experience Evaluation

A user study was conducted to assess usability. Participants were instructed to write in the air in front of the camera, with minimal training required initially. On average, training time was less than 3 minutes. Users encountered minor challenges such as shaky effects and hand stiffness in the initial attempts, but these issues were quickly overcome. Overall, users provided positive feedback and appreciated the functionality of the system.

VI. Results and Discussion

5.1. Parameter Optimization

The model underwent multiple training iterations using a dataset comprising 20,000 trajectories, with 10-fold cross-validation applied. Each iteration consisted of 18,000 training and 2000 validation data points. Various batch sizes and iterations were experimented with to optimize parameters.

The LSTM model was trained for 25, 50, 75, and 100 iterations. Figure X illustrates the variation in accuracy across different iterations and batch sizes. It can be observed that the highest accuracy was achieved with a batch size of 256 for 100 iterations. Therefore, optimal parameters for LSTM include a batch size of 256 and 100 iterations.

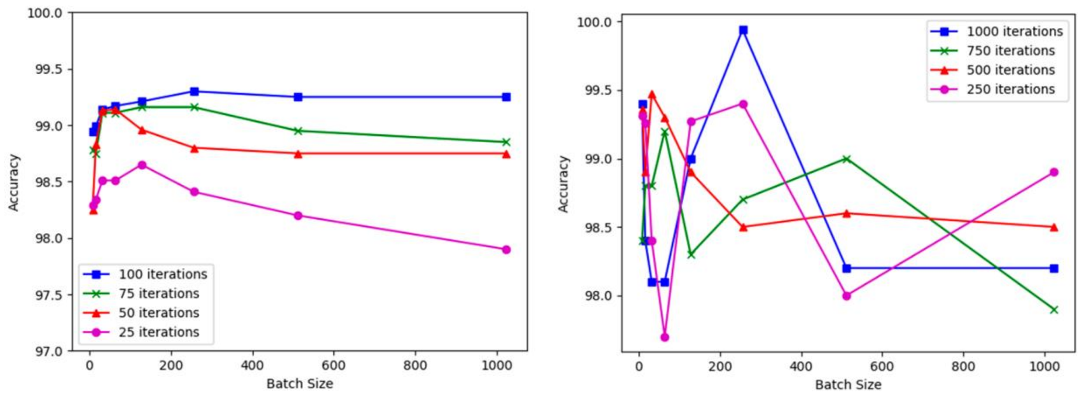


Figure Parameter tuning: (a) batch size vs accuracy for LSTM network, (b) batch size vs accuracy for CNN network. The convergence is faster in LSTM rather than CNN in terms of the number of iterations

In contrast to LSTM, the CNN model underwent 250, 500, 750, and 1000 iterations (as depicted in Figure 10b). It necessitated a higher number of iterations compared to LSTM to reach convergence. The highest and optimal accuracy was achieved with a batch size of 256 and 1000 iterations. Therefore, subsequent discussions will be based on using a batch size of 256 and 100 iterations for LSTM, and 1000 iterations for CNN. The scatterplot in Figure X appears disordered due to the absence of a definitive relationship indicating linear accuracy with respect to iteration and batch size.

Table 1. Evaluation of the Proposed LSTM and CNN Network

Algorithm	Accuracy (%)
CNN + Without Normalization	98.26
CNN + Nearest Neighbor	98.89
CNN + Root Point	98.73
CNN + Nearest Neighbor + Root Point	99.06
LSTM + Without Normalization	98.68
LSTM + Nearest Neighbor	99.08
LSTM + Root Point	99.02
LSTM + Nearest Neighbor + Root Point	99.17

We conducted a comprehensive experiment on the RTD dataset utilizing LSTM and CNN networks. It was observed that the LSTM model achieves satisfactory accuracy within approximately 20 iterations, after which it stabilizes. The accuracy trend over iterations is illustrated in Figure A1 (Appendix A). Notably, a significant peak in accuracy was reached within the first 15 or 20 iterations. Conversely, the CNN model stabilizes around 150 iterations (as depicted in Figure A2 (Appendix A)). These abrupt changes in model loss and accuracy suggest rapid adaptation to the dataset. Both CNN and LSTM models performed admirably. However, the highest accuracy was attained with a fully normalized RTD dataset using LSTM. Further details are presented in Table 2. The poorest performance was observed with non-normalized data for both LSTM and CNN. Additionally, it was noted that the nearest neighbor normalization technique was more influential than the root point normalization method.
The confusion matrix, depicted in Figure A3 (Appendix A), illustrates the performance of both CNN and LSTM models. CNN exhibited a higher false recognition rate compared to LSTM. For instance, the digit 6 was misclassified as 1 sixteen times by CNN, whereas LSTM corrected this automatically. Overall, LSTM outperformed CNN in all scenarios, likely due to its ability to handle long-term dependencies inherent in time-series data such as trajectories. Nevertheless, both CNN and LSTM achieved commendable accuracies of 99.06% and 99.17%, respectively.
Evaluation on the 6DMG dataset provides a more comprehensive assessment of our proposed model, as the RTD dataset lacks benchmarks. A detailed comparison is presented in Table 3. Chen et al. conducted an extensive analysis of the 6DMG dataset [38], considering both user-dependent and user-independent cases, as well as explicit and implicit motions. Their initial model achieved 96.9% accuracy for explicit motion characters using an HMM classifier, which was later improved to 99.2% accuracy [13]. Subsequent approaches, such as the LSTM-based method by Xu and Xue, yielded lower accuracy compared to previous works [32]. In a contemporary study, Yana and Onoye combined BLSTM and CNN features, achieving a maximum accuracy of 99.27% [39]. This comparison underscores the competitive performance of our proposed network.
Table 2. Various Algorithm and there Accuracy

Algorithm	Algorithm	Accuracy (%)
[38]	HMM	96.9
[13]	HMM	99.2
[32]	LSTM	93.98
[39]	BLSTM	99.27
Proposed Method	CNN	99.26
	LSTM	99.32

In this scenario, the proposed LSTM architecture surpasses previous works, achieving a remarkable 99.32% accuracy, the highest reported to date. Similarly, the accuracy of the proposed CNN model aligns closely with [39]. While previous researchers also utilized LSTM layers, our approach of employing consecutive LSTM layers and preprocessing the dataset through normalization led to higher accuracy compared to previous methods.

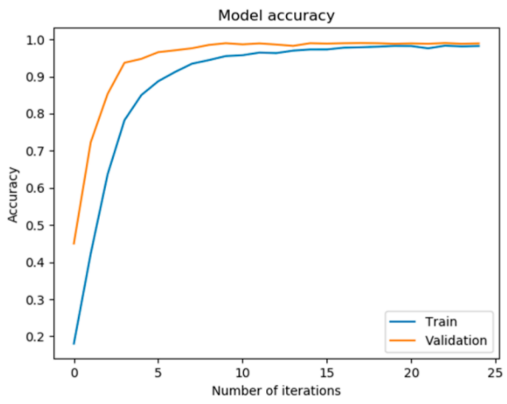
5.2. Result Image

VI. CONCLUSION

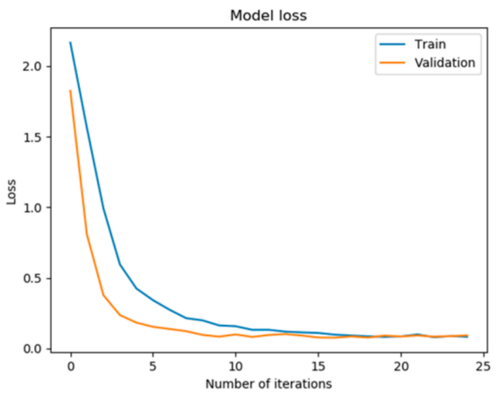
This study presents a trajectory-based air-writing recognition system leveraging an Intel RealSense SR300 camera, accompanied by the development of two deep learning-based algorithms for trajectory recognition. Offering a paperless writing solution, our approach diverges from previous studies reliant on motion sensors and handheld devices, opting instead for a vision-based strategy for improved user experience. Through extensive experimentation under various normalization conditions, we determined that normalized 3D data yielded optimal results. Leveraging the 6DMG dataset, we achieved the highest accuracy reported to date. The primary contribution lies in the design of a network that enhances recognition accuracy and addresses dataset challenges prevalent in current research. The highest recognition accuracies attained for the RTD and 6DMG datasets using CNN are 99.06% and 99.26%, respectively. Notably, the highest accuracy was achieved with LSTM, recording 99.17% and 99.32% for the RTD and 6DMG datasets, respectively. Furthermore, these accuracies were achieved within a reasonable number of iterations, underscoring the efficiency and rapid learning capability of the model. A comparative analysis with prior studies revealed notably higher accuracy in our work. Future endeavors will focus on designing a model tailored for continuous writing systems applicable to augmented reality/virtual reality or gesture-based interfaces.

Appendix A

Here, we present plots illustrating the relationship between iterations, accuracy, and loss for both LSTM and CNN networks. The rapid convergence of these networks in terms of iterations is evident.



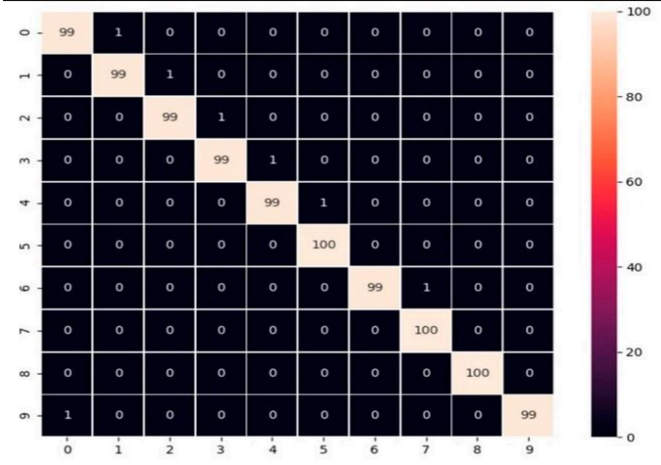
(a)



(b)

Figure A1. Model accuracy and loss for LSTM network: (a) accuracy over 100 iterations. (b) loss over 100 iterations

The confusion matrix for LSTM and CNN network. Itshows that the accuracy is a little higher in LSTM than CNN. However, the difference is not significant


(a)

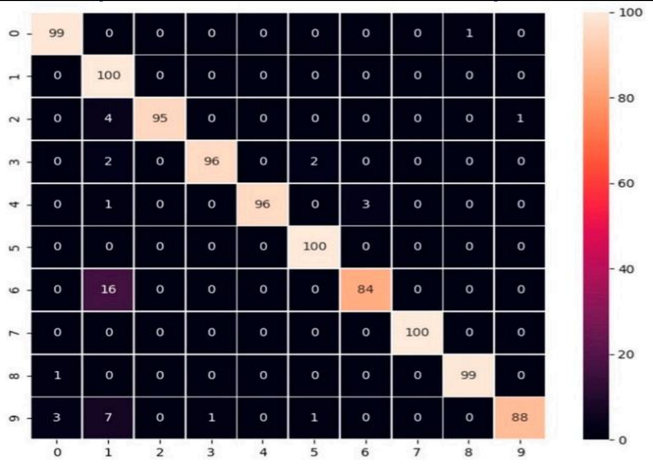

(b)

Figure A2. Confusion matrix for RTD dataset using: (a) LSTM and (b) CNN networks. The false recognition rate for CNN is higher than for LSTM.

REFERENCE

1. Welch, G.; Foxlin, E. Motion tracking: No silver bullet, but a respectable arsenal. IEEE Comput. Graph. Appl. 2002, 22, 24–38.
2. Liu, H.; Wang, L. Gesture recognition for human-robot collaboration: A review. Int. J. Ind. Ergon. 2018, 68, 355–367.
3. Modanwal, G.; Sarawadekar, K. Towards hand gesture based writing support system for blinds. Pattern Recognit. 2016, 57, 50–60.
4. Frolova, D.; Stern, H.; Berman, S. Most Probable Longest Common Subsequence for Recognition of Gesture Character Input. IEEE Trans. Cybern. 2013, 43, 871–880.
5. De, O.; Deb, P.; Mukherjee, S.; Nandy, S.; Chakraborty, T.; Saha, S. Computer vision based framework for digit recognition by hand gesture analysis. In Proceedings of the 2016 IEEE 7th Annual Information Technology, Electronics and Mobile Communication Conference, Vancouver, BC, Canada, 13–15 October
6. Poularakis, S.; Katsavounidis, I. Low-Complexity Hand Gesture Recognition System for Continuous Streams of Digits and Letters. IEEE Trans. Cybern. 2016, 46, 2094–2108.
7. Qu, C.; Zhang, D.; Tian, J. Online Kinect Handwritten Digit Recognition Based on Dynamic Time Warping and Support Vector Machine. J. Inf. Comput. Sci. 2015, 12, 413–422.
8. Mohammadi, S.; Maleki, R. Air-writing recognition system for Persian numbers with a novel classifier. arXiv 2019, arXiv:1903.03558.
9. Tian, J.; Qu, C.; Xu, W.; Wang, S. KinWrite: Handwriting-Based Authentication Using Kinect. In Proceedings of the 20th Annual Network and Distributed System Security Symposium, San Diego, CA, USA, 24–27 February 2013.
10. Zhang, X.; Ye, Z.; Jin, L.; Feng, Z.; Xu, S. A new writing experience: Finger writing in the air using a kinect sensor. IEEE Multimed. 2013, 20, 85–93.
11. Kumar, P.; Saini, R.; Roy, P.P.; Dogra, D.P. Study of Text Segmentation and Recognition Using Leap Motion Sensor. IEEE Sens. J. 2017, 17, 1293–1301.
12. Zhou, Y.; Dai, Z.; Jing, L. A controlled experiment between two methods on ten-digits air writing. In Proceedings of the 2016 16th IEEE International Conference on Computer and Information Technology, Nadi, Fiji, 8–10 December 2016; pp. 299–302.
13. Chen, M.; AlRegib, G.; Juang, B.-H. Air-Writing Recognition—Part I: Modeling and Recognition of Characters, Words, and Connecting Motions. IEEE Trans. Hum.-Mach. Syst. 2016, 46, 403–413.
14. Wang, J.S.; Chuang, F.C. An accelerometer-based digital pen with a trajectory recognition algorithm for handwritten digit and gesture recognition. IEEE Trans. Ind. Electron. 2012, 59, 2998–3007.
15. Yana, B.; Onoye, T. Air-writing recognition based on fusion network for learning spatial and temporal features. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. 2018, E101A, 1737–1744.