Dean&Francis

ISSN 2959-6157

A 3D Maze Rehabilitation Game for Upper Limb Stroke Recovery Based on MediaPipe and SVM

Roubing Yao

Maynooth International Engineering College, Fuzhou University, Fuzhou, China
832204225@fzu.edu.cn

Abstract:

This study addresses key challenges in upper limb rehabilitation for stroke patients, including the dependence on third-party assistance, the high cost of smart devices, and the lack of multisensory stimulation in traditional methods. This paper proposes a 3D maze rehabilitation game system based on gesture recognition. The system is built on a dual architecture consisting of a Python-based gesture recognition endpoint and a Unity-based game interaction endpoint. To reduce costs, the hardware relies on a standard computer camera, eliminating the need for depth sensors. On the software side, the MediaPipe framework is employed to track 21 3D hand keypoints in real-time. Three feature types, relative coordinates, finger angles, and relative distances, are extracted and normalized to mitigate distance and scale interference. Using publicly available datasets and an Support Vector Machine algorithm, this study developed a model to classify 10 distinct gestures. The system ensures cross-platform communication via UDP, and Unity is used to create dual scenes, fully mapping gestures to game controls, enabling patients to interact with the game through hand gestures during rehabilitation training. Experimental results show that the system achieves 100% accuracy at an optimal recognition distance of 25 cm. However, accuracy decreases for complex gestures when the distance increases to 45 cm or in low-light conditions, due to the impact of image quality. This system offers a low-cost, immersive rehabilitation solution for stroke patients and provides insights into the development of intelligent rehabilitation technologies.

Keywords: Hand rehabilitation; gesture recognition; MediaPipe.

1. Introduction

Stroke is one of the primary long-term causes of disability, affecting approximately 150,000 people each year, among whom 60% of survivors experience upper limb impairment [1]. The fine motor dysfunction of the hand caused by such impairments not only directly undermines patients' autonomy in daily life, but also imposes a heavy physical and psychological burden as well as a caregiving burden on patients and their families [2]. Hand rehabilitation training, which stimulates neural remodeling through repetitive movements, can effectively improve the function of the affected side and autonomy in daily life for hemiplegic patients during the stroke recovery period, thereby helping patients achieve substantial improvements in motor function [1].

From the perspective of rehabilitation technology classification, hand rehabilitation training can be categorized into two types: traditional equipment-assisted training and technology-driven intelligent rehabilitation, yet both have significant limitations. In the traditional model, on the one hand, the high labor costs associated with the one-on-one assistance provided by physical therapists constitute the primary economic burden of conventional rehabilitation [3]. On the other hand, mechanical training devices (e.g., hand grippers) lack progress feedback and incentive mechanisms, resulting in poor patient compliance [4]. Although intelligent rehabilitation technology has offered a new direction for overcoming such predicaments, as demonstrated by machine-assisted training systems like HandTutor, which can enhance grasping function [5], it still suffers from several limitations. Specifically, most machine-assisted training approaches face contradictions between equipment dependence and costs. For example, exoskeleton devices such as RobHand require installation and debugging by professional personnel [6]. In addition to electroencephalography-based brain-computer interfaces, which demand specialized venues and invasive equipment, they are capable of enabling intention-driven training [7]. Furthermore, existing research on rehabilitation robots typically centers on the design optimization of device structures and the realization of rehabilitation functions, either overlooking patients' physical and mental health needs [8] or only providing dual feedback of vision and kinesthesis, which remains a gap with the "multi-sensory collaborative stimulation" needs of patients, as revealed by surveys based on the Kano Mode l [4]. Additionally, although the "active interactive rehabilitation system based on surface electromyographic (sEMG) signals" developed by Yang et al. can assist patients in controlling small-range extension of the hand via sEMG signals and incorporates multimodal feedback, it remains inadequate in terms of motor precision [8].

In recent years, computer vision-based hand gesture recognition technology has emerged as an auspicious direction in hand rehabilitation. By integrating advanced sensing technologies and computer pattern recognition technologies [9], this type of technology enables a more natural and optimal interaction method for human-computer interaction, thereby breaking through some limitations of traditional rehabilitation and robotic rehabilitation. Therefore, this study aims to design and adopt an engaging hand rehabilitation method based on image-based hand gesture recognition. Specifically, using Python as the development base, it employs the MediaPipe machine learning framework for hand gesture detection and semantic definition [10]. For gesture classification, the support vector machine (SVM) algorithm is utilized. In addition, drawing on the user experience design principles derived from the Three-Level Theory of Emotion [4], a 3D maze game was designed in Unity, which provides a new technical pathway for improving the function of the affected side in stroke patients.

2. Method

2.1 Overview of framework

Guided by the hand rehabilitation needs of stroke patients, this study develops a gesture recognition-based 3D maze rehabilitation game system. The overall framework of the system is illustrated in Fig. 1 below. In terms of hardware, a computer camera is used as the input device. This study implements a dual-end architecture consisting of a "Python-based gesture recognition terminal" and a "Unity game interaction terminal."

Based on Python, images are captured using OpenCV, and Google's open-source MediaPipe framework is employed as the core tool for hand key point detection. This enables real-time, accurate tracking of 21 hand key points. The SVM algorithm is utilized for gesture classification, addressing the multi-category gesture recognition challenge in small-sample scenarios. Additionally, the UDP protocol is specified as a cross-platform data transmission solution. Upon receiving the data, the Unity terminal maps ten types of gestures to corresponding game control commands.

ISSN 2959-6157

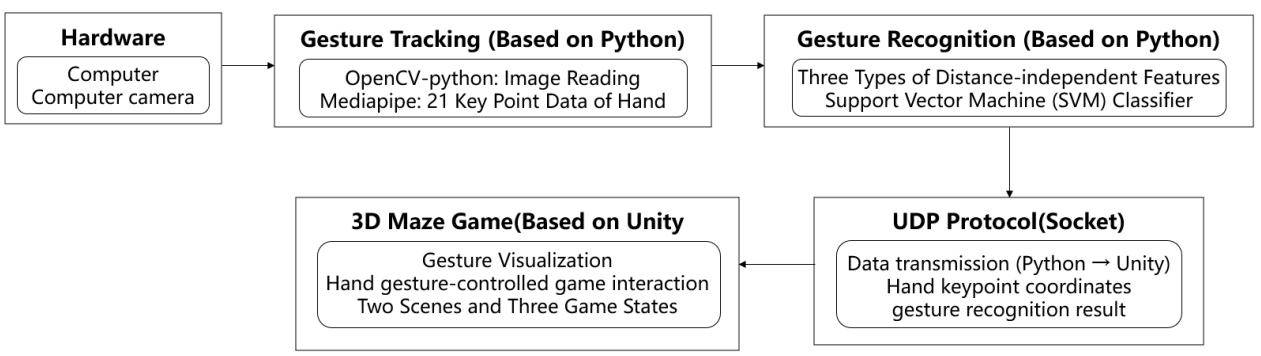

Fig. 1 Flowchart of the framework (Picture credit: Original)

2.2 Data preparation

This study utilizes the Sign Language Digits Dataset on GitHub, provided by students from Turkey Ankara Ayrancı Anadolu High School [11]. The dataset consists of images representing 10 classes of sign language digits (0-9), with each image having a resolution of 100 x 100 pixels and an RGB color space, as illustrated in Fig. 2. The dataset includes samples from 218 students, with each student contributing 10 samples. This dataset is designed to support gesture recognition tasks, and Fig. 2 shows some sample cases.

After performing preliminary processing on the sample data based on brightness and contrast, the dataset was analyzed for gesture recognition. The results show that gesture 0 is recognized in 130 images, gesture 1 in 144 images, gesture 2 in 110 images, gesture 3 in 202 images, gesture 4 in 161 images, gesture 5 in 207 images, gesture 6 in 131 images, gesture 7 in 132 images, gesture 8 in 162 images, and gesture 9 in 195 images.

Fig. 2 Examples of Sign Language Digits Dataset [11]

2.3 MediaPipe

MediaPipe is an open-source framework developed by Google Research for graphical data processing. It supports cross-platform development and provides multi-language interfaces [12], serving as an efficient tool for the development of multimedia machine learning applications.

Its framework architecture comprises three core components: a framework for sensory data inference, which coordinates data input, model inference, and result output, a performance evaluation toolset, which provides a quantitative basis for system optimization, and a reusable inference and processing component library, which supports flexible combination to adapt to diverse requirements [13].

In the core hand recognition module, MediaPipe Hands, a two-stage model of "palm detection and keypoint estimation" is used to achieve real-time, precise tracking of hand skeletons in RGB images [14]. MediaPipe leverages advanced computer vision algorithms to efficiently recognize and track 21 key points of the hand in real-time, as depicted in Fig. 3. This enables more diverse interactions between game characters and players without the need for any input devices (such as a keyboard or mouse) [15], significantly enhancing the immersion and interactivity of the game. Its modular architecture, cross-platform compatibility, real-time performance, and accuracy make it highly suitable for the gesture recognition requirements of this study, establishing the core technological foundation for the development of a 3D maze game for stroke rehabilitation.

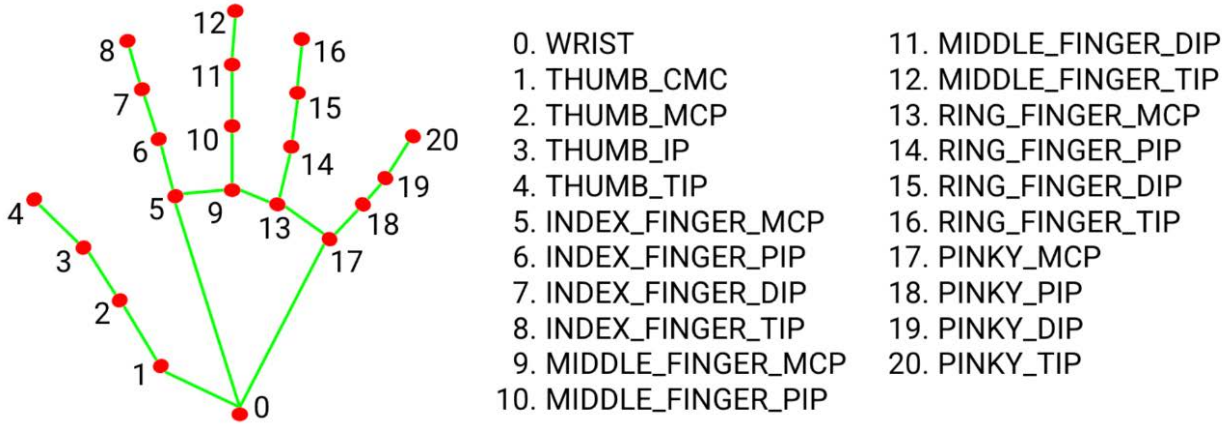

Fig. 3 The 21 keypoints of a single hand detected by MediaPipe [16].

2.4 SVM

SVM is a supervised machine learning algorithm grounded in statistical learning theory. The core principle of SVM is to identify the optimal hyperplane within the feature space that maximizes the margin between two categories of samples. This hyperplane enables the linear separation of the samples and is determined by a limited number of "support vectors," which guarantees a global optimal solution for classification, thereby ensuring the accuracy of the target detection classifier [17].

For multi-class gesture classification, SVM constructs a multi-layer classification structure through binary classification iterations, breaking the multi-class problem into a series of binary subtasks for incremental sample partitioning [18]. Additionally, kernel functions are utilized to map non-linearly separable samples from a lower-dimensional space to a higher-dimensional feature space, where the optimal hyperplane is identified to achieve classification. The classification performance of SVM is highly dependent on the choice of kernel function and its parameters: the more refined the kernel function's representation of the sample features, the more accurate its depiction of the sample's surface similarity. However, this must be balanced with the associated computational complexity [14].

In the context of gesture classification, SVM offers several key advantages: first, its global optimal solution ensures high classification accuracy; second, its robust capability in handling small sample sizes, nonlinear data, and high-dimensional feature spaces makes it well-suited for gesture recognition applications, where data collection is challenging and feature dimensions are typically large.

Since this study utilizes a standard computer camera rather than a depth camera, depth information cannot be acquired. To mitigate the interference of hand-to-camera distance and scale variations on recognition results, this study designs and implements a feature normalization process. The core of this process involves extracting three types of scale-invariant, robust features—"relative coordinates, finger angles, and relative distances"—from the 21 3D hand keypoints output by MediaPipe. These three feature types total 89 dimensions: 63 dimensions for relative coordinates, 5 dimensions for finger angles, and 21 dimensions for relative distances. The specific implementation sequence is as follows: normalization of relative coordinates, calculation of finger bending angles, and computation of relative distances. Based on this, an SVM classification model is built using the scikit-learn library to achieve high-accuracy gesture classification.

2.5 User Datagram Protocol (UDP)

The selection of cross-end data transmission protocols must prioritize real-time performance and stability as the primary criteria. In this project, UDP is employed to establish a bidirectional communication link between the Python-based gesture recognition terminal and the Unity

ISSN 2959-6157

game terminal. Among the common transport layer protocols, UDP and Transmission Control Protocol (TCP) are the two core options. TCP is a connection-oriented protocol that ensures reliability through connection establishment, data acknowledgment, and flow control mechanisms. However, its inherent redundancy results in transmission delays. In contrast, UDP's connectionless nature eliminates these redundant operations, offering significantly higher transmission efficiency. Additionally, its lightweight design reduces the risk of link congestion in Python-Python and Unity cross-end interactions, thereby demonstrating superior communication stability [19].

For the specific implementation, the Python terminal constructs a UDP client based on the socket library, encapsulating core data such as timestamps, coordinates of 21 hand keypoints, gesture categories, and confidence levels into JSON format, which are subsequently sent to the Unity terminal via a custom send_data function. Concurrently, the Unity terminal implements a UDP communication module using the System.Net.Sockets namespace in C# to receive and parse the transmitted data.

2.6 Unity

To achieve adequate rehabilitation training for patients' hands, this study established two core scenes (main menu and game) in the Unity terminal based on 10 categories of gesture data (0-9) recognized by the Python terminal, and defined three interaction states (game, menu, and victory). Thereby, the full mapping between the 10 gesture categories and game control commands is realized, ensuring the comprehensive engagement of hand movements during the rehabilitation training process. The specific gesture mappings and their corresponding functions are detailed in Table 1.

Table 1. Mapping of gestures to corresponding game interaction functions

Scene & State	Gesture	Function	Description
Main Menu Scene	Gesture 8	Start Game	Switch to the game scene
	Gesture 7	Quit Game	Close the Unity editor or application
Game Scene (Default State)	Gesture 1	Move Forward	Control the character to move forward
	Gesture 2	Move Backward	Control the character to move backward
	Gesture 4	Move Left	Control the character to move left
	Gesture 5	Move Right	Control the character to move right
	Gesture 3	Open Menu	Display the in-game menu
Game Scene (Menu State)	Gesture 0	Return to Main Menu	Switch to the main menu scene
	Gesture 9	Resume Game	Close the menu and resume the game
	Gesture 6	Restart Game	Close the menu and restart the game
Game Scene (Victory State)	Gesture 7	Quit Game	Close the Unity editor or application
	Gesture 6	Restart Game	Close the menu and restart the game

Based on the aforementioned design concepts, this study developed a lightweight 3D maze rehabilitation training game. In this game, patients are required to manipulate the game avatar through gesture-driven interaction, moving it to a predetermined target point to achieve the victory criterion. Thereby, they complete fine hand movement training in a gamified experience.

3. Result and Discussion

3.1 The implementation of the gesture recognition game

This study ultimately completed the development and deployment of a gesture recognition game in the Unity engine. The core interaction logic of the game requires patients to control the movement of the game entity through predefined specific gestures, aiming to successfully reach the preset endpoint in the maze scene, which is considered as winning the game. The running state of the game within the Unity editor is illustrated in Fig. 4.

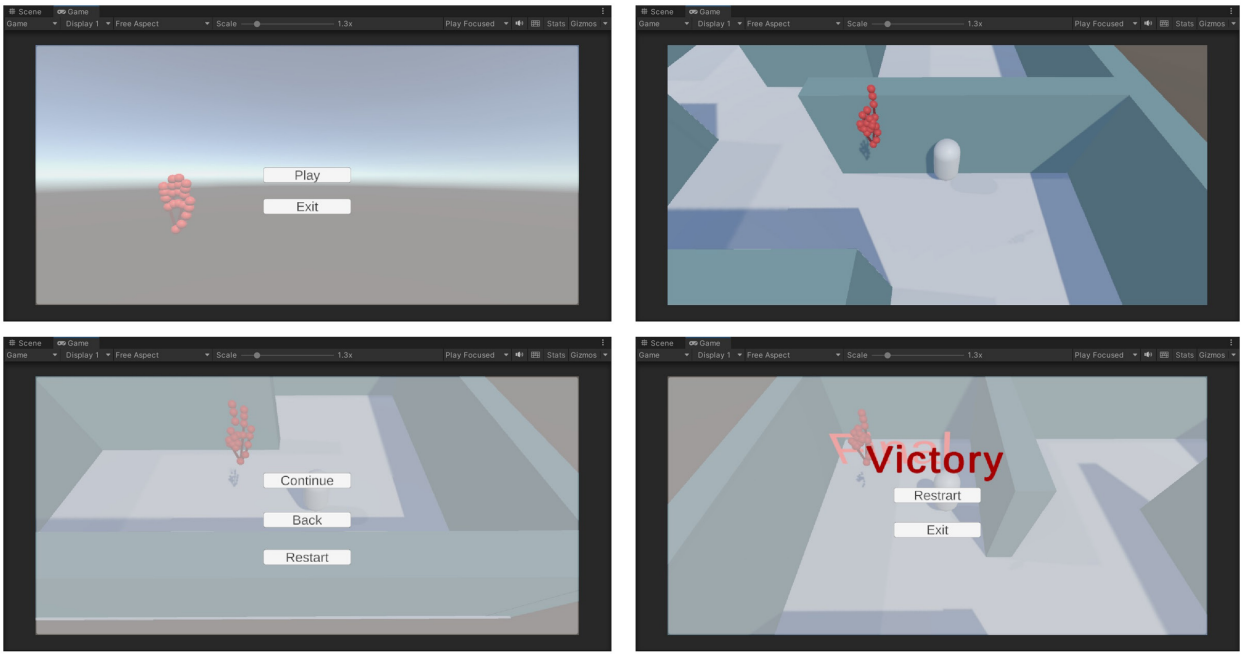

Fig. 4 Screenshots of the game (Picture credit: Original)

To validate the system's real-time performance, the research tested the data transmission latency between the Python gesture recognition module and the Unity game module. Multiple repeated tests were conducted in a Windows 11 system environment with an AMD Ryzen 7 5800H processor. Testing results confirm that the average transmission delay of this communication link is approximately 11 ms, which satisfies the real-time synchronization requirements for gesture movements and game interactions.

3.2 Impact of distance to the camera on gesture recognition accuracy

This study utilizes a computer-built-in camera instead of a depth camera, which limits the acquisition of hand depth information. As a result, variations in distance and scale between the hand and the camera may introduce interference, potentially affecting recognition accuracy. While feature normalization processing has been incorporated into the real-time gesture recognition workflow to mitigate such interference, the issue has not been entirely resolved. To quantitatively analyze the impact of camera distance on recognition performance, a controlled experiment was designed. Based on the pre-experimental determination of the effective recognition distance boundary, gesture recognition accuracy tests were conducted at distances of 25 cm, 35 cm, and 45 cm from the camera, with a sample size of 50 repetitions for each gesture category, and a confidence threshold of 0.5 was set for the gesture recognition module. The gesture recognition images at these distances are illustrated in Fig. 5.

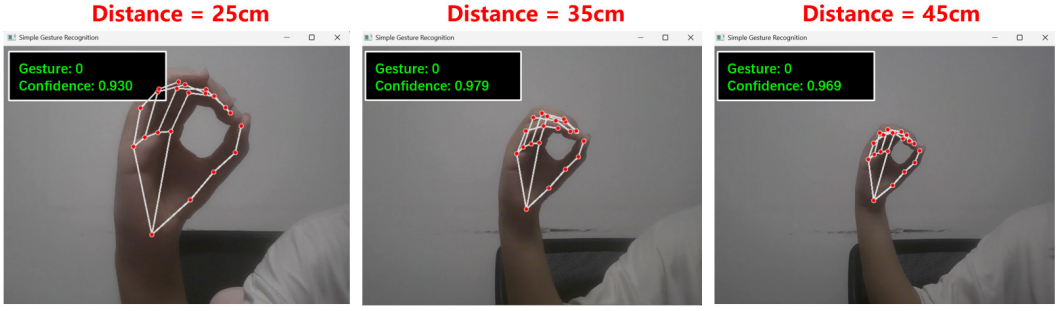

Fig. 5 Gesture recognition tests at different distances (Picture credit: Original)

ISSN 2959-6157

Table 2. Recognition accuracy of various gesture categories under different camera distances

Gesture Category	Recognition Accuracy(100%)		
	Distance = 25	Distance = 35	Distance = 45
Gesture 0	100%	100%	100%
Gesture 1	100%	100%	96%
Gesture 2	100%	100%	96%
Gesture 3	100%	88%	8%
Gesture 4	100%	100%	100%
Gesture 5	100%	68%	12%
Gesture 6	100%	100%	76%
Gesture 7	100%	100%	72%
Gesture 8	100%	90%	10%
Gesture 9	100%	100%	88%

Experimental data presented in Table 2 indicate that the distance between the hand and the camera significantly impacts recognition accuracy, with different gesture categories exhibiting distinct sensitivities to variations in distance. The optimal recognition distance of the system is 25 cm, within which 100% recognition accuracy can be achieved for all gesture categories. When the distance increases to 35 cm, the recognition robustness of the system for complex gestures (e.g., Gesture 3 and Gesture 5) decreases significantly, necessitating targeted optimization of feature extraction strategies to enhance robustness. When the distance increases to 45 cm, the overall recognition accuracy of the system drops sharply. This phenomenon suggests that when the distance between the hand and the camera exceeds a specific threshold, the pixel proportion of the hand in the image decreases significantly, leading to a substantial increase in the detection error of MediaPipe hand key points. Even with feature normalization, it is difficult to effectively offset the interference caused by scale variations.

3.3 Impact of lighting conditions to the camera on gesture recognition accuracy

To further investigate the impact mechanism of ambient light intensity, a critical interfering factor, on gesture recognition accuracy, this study designed and conducted a controlled experiment under two distinct lighting conditions: well-lit and dimly lit environments.

Based on the findings presented in Section 3.1, Impact of distance to the camera on gesture recognition accuracy, this study confirmed that 25 cm serves as the system's optimal recognition distance. At this distance, interference arising from distance and scale variations is negligible, with all gesture categories attaining 100% baseline accuracy. Accordingly, 25 cm was selected as the fixed quantitative testing distance for the present experiment. The experiment encompassed 10 gesture categories (0–9), with 50 test samples per category. A confidence threshold of 0.5 was set for the gesture recognition module, and the specific parameters of the lighting conditions are detailed in Fig. 6.

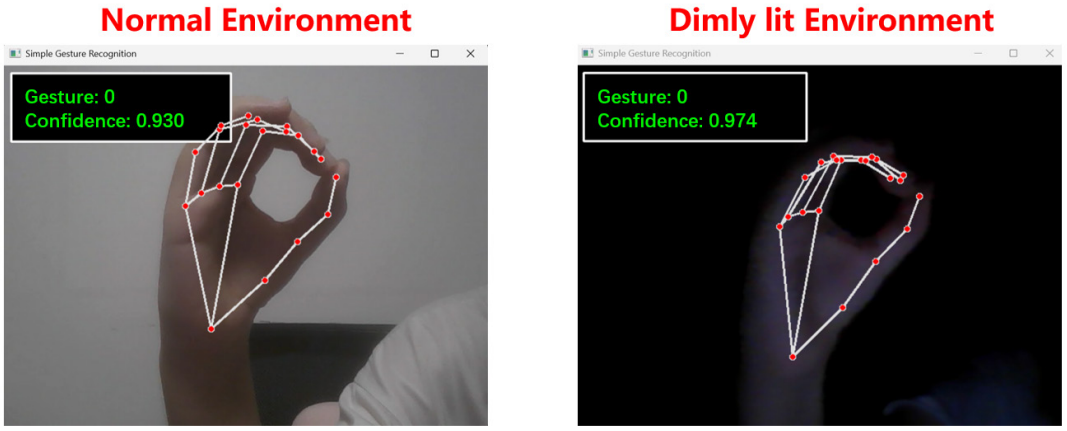

Fig. 6 Gesture recognition tests under different lighting conditions (Picture credit: Original)

Table 3. Gesture recognition accuracy under different lighting conditions

Testting Conditions	Recognition Accuracy(100%)
Normal (Distance = 25)	100%
Dimly (Distance = 25)	82%

Experimental results in Table 3 demonstrate that lighting conditions have a significant impact on the gesture recognition accuracy of the system. A detailed analysis of the experimental outcomes reveals that the underlying mechanism of this impact is the degradation of image quality in dimly lit environments. Insufficient ambient light intensity leads to a notable reduction in the Signal-to-Noise Ratio of images in the hand region, resulting in blurred finger edges and texture details. This degradation contributes to two primary types of errors in the MediaPipe hand detection module. The first is hand target misdetection, where the hand region fails to be effectively located in the image. The second is keypoint tracking drift, which occurs when the system is unable to continuously and accurately capture the hand movement trajectory. Both types of errors lead to the distortion of critical robust features, such as relative coordinates and finger angles, during the feature extraction stage, ultimately causing a decrease in recognition accuracy. These error states are illustrated in Fig. 7.

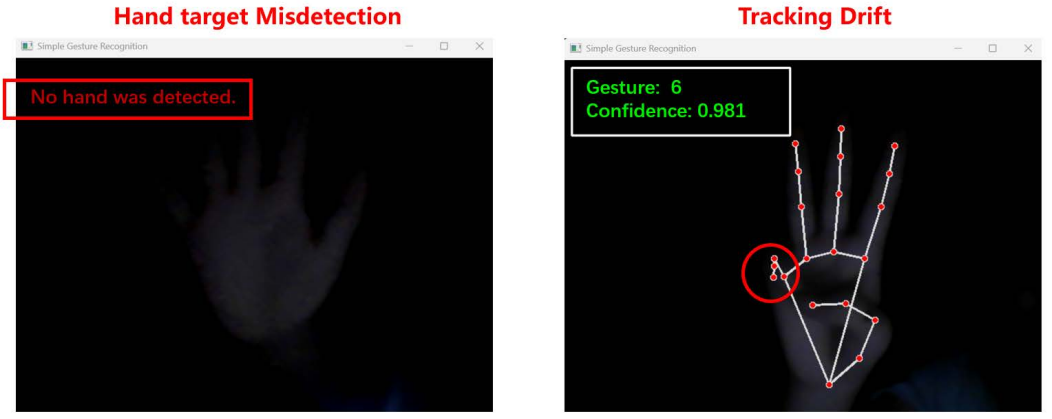

Fig. 7 Two types of errors in gesture recognition under dim lighting conditions (Picture credit: Original)

4. Conclusion

This study addresses key limitations in upper limb rehabilitation for stroke patients, including the reliance of traditional methods on third-party assistance, the high cost of smart devices, and the lack of multisensory stimulation. This paper developed a gesture-recognition-based 3D maze rehabilitation game system.

The system is built on a dual architecture of Python and Unity. It tracks 21 hand keypoints using MediaPipe and implements classification of 10 gesture types through a combination of publicly available datasets and a Support Vector Machine (SVM) algorithm. Cross-platform communication is facilitated via the UDP protocol, and gesture-to-game control mapping is fully implemented within Unity. Experimental results indicate that the system achieves 100% accuracy at a recognition distance of 25 cm; however, accuracy decreases for complex gestures at a distance of 45 cm or under low-light conditions.

Current limitations of the system include a single game scenario and the absence of a quantitative rehabilitation assessment. Future work will focus on enriching game content, incorporating quantitative measures such as finger joint angle analysis, and improving recognition accuracy in complex environments through the introduction of deep learning models, thereby providing patients with more precise and personalized rehabilitation solutions.

References

[1] Xu Jiayun, Zhou Liping. Effect of finger fine motor training on hand function in stroke convalescent patients with hemiplegic hand dysfunction. Chinese Journal of Modern Drug Application (in Chinese), 2024, 18(17): 156-159. DOI:10.14164/j.cnki.cn11-5581/r.2024.17.043.

[2] Yan Cheng, Chen Cheng, Zhang Xiang, et al. Design of hand rehabilitation training system based on mirror neuron theory. Chinese Medical Equipment Journal (in Chinese), 2021, 42(01): 26-31. DOI:10.19745/j.1003-8868.2021005.

[3] Yeng, Angelina Chow Mei, et al. Hand gesture controlled

ISSN 2959-6157

game for hand rehabilitation. Proceedings of the International Conference on Computer, Information Technology and Intelligent Computing (CITIC 2022). Amsterdam, The Netherlands: Atlantis Press, 2022.

[4] Zhang Xiaoxuan, Chen Xiang. Design practice of hand rehabilitation training products based on Kano-Pugh. Packaging Engineering (in Chinese), 2024, 45(22): 76-83. DOI:10.19554/j.cnki.1001-3563.2024.22.008.

[5] Bayındır, Ozun, Gülseren Akyüz, and Nimet Sekban. The effect of adding robot-assisted hand rehabilitation to conventional rehabilitation program following stroke: A randomized-controlled study. Turkish Journal of Physical Medicine and Rehabilitation, 2022, 68(2): 254.

[6] Barria P, Riquelme M, Reppich H, et al. Hand rehabilitation based on the RobHand exoskeleton in stroke patients: A case series study. Frontiers in Robotics and AI, 2023, 10: 1146018.

[7] Baniqued, Paul Dominick E., et al. Brain–computer interface robotics for hand rehabilitation after stroke: a systematic review. Journal of NeuroEngineering and Rehabilitation, 2021, 18(1): 15.

[8] Li Yang, Chen Enwei, Wu Di, et al. Research on hand rehabilitation interaction system based on perceptual feedback. Journal of Hefei University of Technology (Natural Science Edition) (in Chinese), 2025, 48(05): 583-590.

[9] Xie Yinggang, Wang Quan. Survey of dynamic hand gesture recognition based on vision. Computer Engineering and Applications (in Chinese), 2021, 57(22): 68-77.

[10] Meng Jie, Yang Pengcheng, Yang Zhao, et al. Design of natural gesture interaction system for phantom imaging device based on Mediapipe. Foreign Electronic Measurement Technology (in Chinese), 2023, 42(03): 116-122. DOI:10.19652/j.cnki.femt.2204392.

[11] Mavi, A. A New Dataset and Proposed Convolutional Neural Network Architecture for Classification of American Sign Language Digits. arXiv preprint, 2020: arXiv:2011.08927 [cs.CV].

[12] Liu Defa. Digital gesture recognition based on MediaPipe. Electronic Production (in Chinese), 2022, 30(14): 55-57. DOI:10.16589/j.cnki.cn11-3571/tn.2022.14.015.

[13] Lugaresi, Camillo, et al. Mediapipe: A framework for perceiving and processing reality. Third Workshop on Computer Vision for AR/VR at IEEE Computer Vision and Pattern Recognition (CVPR), 2019.

[14] Lin Zhitao. Research on Quantitative Assessment System for Hand Virtual Rehabilitation Based on Dynamic Gesture Recognition. Master's thesis, Nanchang University (in Chinese), 2024. DOI:10.27232/d.cnki.gnchu.2024.001298.

[15] Islam M R, Rahman R, Ahmed A, et al. NFS: A hand gesture recognition based game using MediaPipe and pygame. arXiv preprint, 2022: arXiv:2204.11119.

[16] Google AI. MediaPipe Hand Landmark Detection. [Online]. Available: https://ai.google.dev/edge/mediapipe/solutions/vision/hand_landmarker?hl=zh-cn

[17] Fu Zhikai, Li Wenxin, Luo Xinkui. A survey of dynamic gesture recognition technology based on vision. Computer Measurement & Control (in Chinese), 2025, 33(01): 9-19. DOI:10.16526/j.cnki.11-4762/tp.2025.01.002.

[18] Abdullah, Dakhaz Mustafa, and Adnan Mohsin Abdulazeez. Machine learning applications based on SVM classification a review. Qubahan Academic Journal, 2021, 1(2): 81-90.

[19] Abbass, Mahmoud Abdelkader Bashery, and Hyun-Soo Kang. Drone elevation control based on python-unity integrated framework for reinforcement learning applications. Drones, 2023, 7(4): 225.