Virtual Reality Interaction System for Rail Transit Emergency Response

Xianghao Wang[1,2], Li Wang[1,2](✉), and Xinyi Du[1,2]

[1] Beijing Engineering Research Center of Urban Traffic Information Intelligent Sensing and Service Technologies, Beijing 100044, China
{xianghaow,wangli,duxinyi}@bjtu.edu.cn

[2] School of Traffic and Transportation, Beijing Jiaotong University, Beijing 100044, China

Abstract. In the field of rail transportation, the application of virtual reality devices is becoming increasingly popular. However, traditional controllers are not convenient to use for on-site operations, and the use of gesture recognition technology for human-computer interaction can effectively solve this problem. This paper proposes a virtual reality interaction system designed specifically for emergency rescue in rail transportation and focuses on introducing its human-computer interaction module for the virtual reality terminal device. The module uses gesture recognition technology to achieve virtual mouse, virtual keyboard, and volume control functions. Experimental results show that the system has superior performance and is suitable for virtual reality device interaction in the rail transportation field. This study provides theoretical basis and practical reference for the design and implementation of virtual reality interaction systems for emergency rescue in rail transportation.

Keywords: Rail transportation · Human-computer interaction · Gesture recognition · Virtual reality

1 Introduction

Virtual reality devices, particularly VR glasses, are widely adopted in the production industry, including rail transportation, for various applications such as virtual map navigation, maintenance guidance, intelligent inspection, emergency rescue, safety training, and education.It has become a trend with great development prospects [1–8].

The rail industry requires more interactive functions in virtual reality devices [3], leading to the emergence of new human-machine interaction methods like gesture and facial recognition [9, 10]. Gesture recognition is becoming mainstream [11] but needs improved sensitivity for better operational experience.

This paper proposes a VR-assisted system for emergency rescue in rail transportation, consisting of a multi-source information fusion system, emergency dispatch instruction system, and a VR-based human-machine interaction system. The human-machine interaction system utilizes gesture recognition for VR/AR scenarios, offering functions like virtual mouse, keyboard, and media control. It enhances user experience, frees the hands of on-site operators, and provides a versatile solution for emergency rescue in rail transportation.

2 Overview of the Virtual Reality-Assisted System for Emergency Rescue in Rail Transportation

The virtual reality-assisted system for emergency rescue in rail transportation consists of three major parts: a field situation awareness system based on multi-source information fusion, an emergency dispatch instruction receiving system, and a human-machine interaction system for virtual reality terminal devices, the structure of the system is shown in Fig. 1. This chapter will provide a detailed explanation of the composition and functions of these three subsystems.

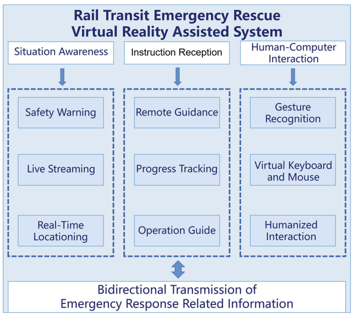
Rail Transit Emergency Rescue
Virtual Reality Assisted System
Situation Awareness
Instruction Reception
Human-Computer Interaction
Safety Warning
Live Streaming
Real-Time Locationing
Remote Guidance
Progress Tracking
Operation Guide
Gesture Recognition
Virtual Keyboard and Mouse
Humanized Interaction
Bidirectional Transmission of Emergency Response Related Information

Fig. 1. The structure of the Rail Transit Emergency Rescue Virtual Reality Assisted System

2.1 Field Situation Awareness System Based on Multi-source Information Fusion

The field situation awareness system collects videos, audio, environment videos, and positioning coordinates using sensors installed on virtual reality devices. This information is then transmitted wirelessly to the emergency command platform. The command platform uses its computing power to understand and analyze the multi-modal information, enabling intelligent perception of the emergency situation on the scene.

2.2 Emergency Dispatch Instruction Receiving System

The system sends dispatch instructions to on-site personnel's virtual reality terminals, which receive and display the instructions. It includes a progress feedback function for operators to confirm completion status, and the terminals also send updates to the command platform for tracking.

2.3 Human-Machine Interaction System for Virtual Reality Terminal Devices

This paper explores the human-machine interaction system for virtual reality devices, using the mediapipe visual recognition module. It surpasses physical controllers, offering an immersive and natural experience with three functional modules: virtual mouse, virtual keyboard, and volume control.

3 Implementation Method of Human-Computer Interaction System for Virtual Reality Terminal Devices

This system has three modules: virtual mouse, virtual keyboard, and volume control. It uses Mediapipe for gesture recognition and explains the architecture and working principle.

3.1 Introduction to the Mediapipe Model

Mediapipe Hands is a lightweight gesture recognition module in the Mediapipe deep learning framework [12, 13]. It can infer 3D coordinates of 21 hand keypoints in real time on mobile devices. The module utilizes a two-stage process of hand detection and gesture recognition, providing accurate depth information and rendering the image with marked keypoints. Mediapipe Hands can predict the depth information and generate the 3D coordinates of the hand keypoints as shown in the Fig. 2. The detailed description of the system structure of the Mediapipe Hands module is provided in the Fig. 3.

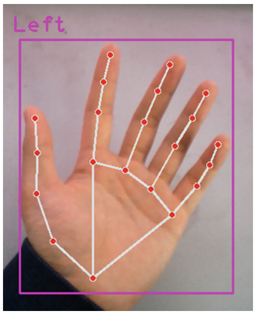

Fig. 2. Display of hand keypoint positions

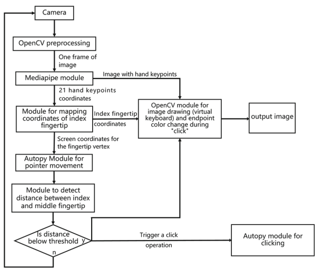
Camera
OpenCV preprocessing
One frame of image
Mediapipe module
Image with hand keypoints
21 hand keypoints coordinates
Module for mapping coordinates of index fingertip
Index fingertip coordinates
OpenCV module for image drawing (virtual keyboard) and endpoint color change during "click"
output image
Screen coordinates for the fingertip vertex
Autopy Module for pointer movement
Module to detect distance between index and middle fingertip
Is distance below threshold
y
n
Trigger a click operation
Autopy module for clicking

Fig. 3. Flowchart for simulating mouse function with hand gestures

Using the real-time hand keypoint coordinate information output by Mediapipe Hands, the spatial position of the hand can be dynamically tracked. In this way, in a mixed reality scene, more natural gesture interaction experience can be achieved based on the real-time position, direction, and shape of the hand.

3.2 Gesture Recognition for Implementing Virtual Mouse Function

Design scheme. To replace the physical mouse, hand gesture recognition analyzes camera images to obtain finger coordinates, mapping them to control the mouse pointer on the screen.

$$\begin{cases} x_2 = x_1 + wSlope * (x_1 - 0) \\ y_2 = y_1 + hSlope * (y_1 - 0) \\ wSlope = \dfrac{wScr - 0}{wCam - 0} \\ hSlope = \dfrac{hScr - 0}{hCam - 0} \end{cases} \tag{1}$$

The mathematical formula for one-dimensional linear interpolation is shown in Eq. (1). In this equation, (x_2, y_2) denote the screen coordinates of the mouse pointer, while (x_1, y_1) represent the coordinates of the top keypoint on the index finger captured by the camera. *wScr* and *hScr* refer to the width and height of the screen; *wCam* and *hCam* refer to the width and height of the camera's captured image. To perform one-dimensional linear interpolation, the NumPy library provides the function. Taking the first equation mentioned earlier as an example, this function interpolates between two arrays. Specifically, given a set of coordinates x_1 to be mapped, the range of values for the points to be mapped is represented by the array [0, 1, ..., *wCam*], while the corresponding range of mapped values is contained in the other array [0, 1, ..., *wScr*].

The autopy module uses finger movement coordinates to control the mouse pointer on the screen, allowing it to simulate mouse movement and clicks based on the distance between finger tips. Figure 3 shows the flowchart of simulating mouse movement with gestures.

Optimization of Pointer Jitter. To reduce finger tremors caused by gesture recognition errors in Mediapipe. We optimized the code to enhance user experience by smoothing mouse movement using a calculation based on the difference between current and previous coordinates. The formula used is as follows:

$$\begin{aligned} clocX &= plocX + (x_2 - plocX)/s \\ clocY &= plocY + (y_2 - plocY)/s \end{aligned} \tag{2}$$

This formula uses several variables. *ClocX* and *clocY* represent the current horizontal and vertical locations of the mouse pointer on the screen, respectively, after optimization. In contrast, *plocX* and *plocY* represent the previous frame's horizontal and vertical locations of the mouse pointer on the screen. x_2 and y_2 denote the current mouse pointer's horizontal and vertical coordinates on the screen before optimization. The smoothening variable controls the balance between pointer smoothing and delay, with a higher value increasing smoothing but also introducing more delay.

Implementation Results. Enabling the feature displays a camera monitoring window with a green box denoting the active finger movement area. The index finger controls the interface pointer, while both the index and middle fingers trigger a click by bringing them close until the end-point between them changes color (Fig. 4).

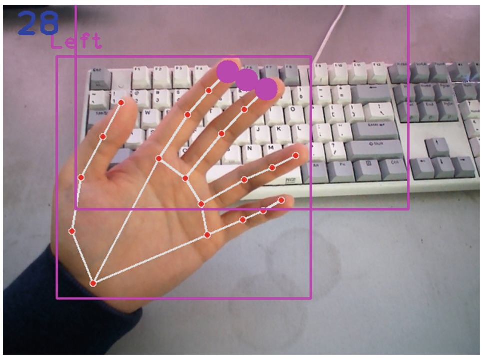

Fig. 4. Image demonstrating virtual mouse controlled by hand gestures

3.3 Virtual Keyboard Input Module

Design Scheme. After capturing the video, the OpenCV preprocessing module converts the image to RGB color space and applies horizontal mirroring. The image is then passed to the Mediapipe module for hand recognition, which returns hand keypoint coordinates and an annotated image. The Finger Tip Matching module matches fingertip coordinates with virtual keyboard buttons and highlights the corresponding button. The Thumb Tip Distance Judgment module detects the distance between the thumb tip and the middle finger and triggers a click if it's below a set threshold. The letter corresponding to the button is sent to the character input module for simulated keyboard input (Fig. 5).

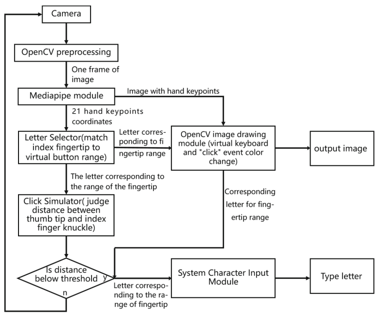
Camera
OpenCV preprocessing
One frame of image
Mediapipe module
Image with hand keypoints
21 hand keypoints coordinates
Letter Selector(match index fingertip to virtual button range)
Letter corres-ponding to fi ngertip range
OpenCV image drawing module (virtual keyboard and "click" event color change)
output image
The letter corresponding to the range of the fingertip
Corresponding letter for fing-ertip range
Click Simulator(judge distance between thumb tip and index finger knuckle)
Is distance below threshold
y
n
Letter correspo-nding to the ra-nge of fingertip
System Character Input Module
Type letter

Fig. 5. Flowchart for controlling virtual keyboard input with hand gestures

Optimization of Clicking Problem. The code was optimized to detect the time interval between consecutive clicks to address the issue of multiple clicks being triggered by a single finger click. The specific implementation method is as follows:

$$cTime - pTime > t \tag{3}$$

The formula uses two variables related to mouse clicks. *cTime* denotes the time when the current click occurred, and *pTime* denotes the time when the previous click occurred, *t* denotes the minimum time interval for triggering a click. The click instruction code runs only when the time interval between the two clicks is greater than *t*. If the interval is less than *t*, it is considered as a single click event; when the interval is greater than *t*, it is considered as two separate clicking operations.

Experimental Results. Figure 6 demonstrates the gesture-controlled media volume program, enabling users to select buttons by touching them with their index finger. A click is triggered by briefly touching the middle finger's second node with the thumb, allowing input on a virtual keyboard for the system's text document.

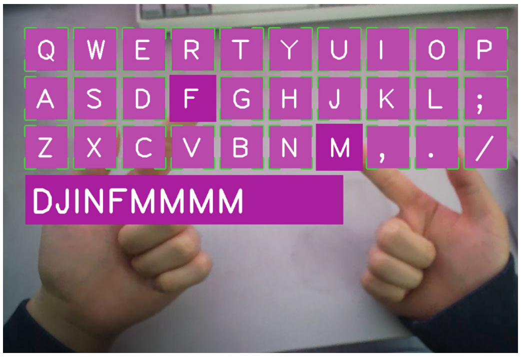

Fig. 6. Image demonstrating virtual keyboard input controlled by hand gestures

3.4 Gesture Recognition for Media Volume Control

Experimental Plan. Mediapipe recognizes hand keypoints, thumb-index finger tip distance is calculated, and the volume control module adjusts the system volume based on the distance. OpenCV draws the image on the screen, achieving real-time video effects at 25 frames per second. Figure 7 shows the flowchart for gesture-controlled media volume control.

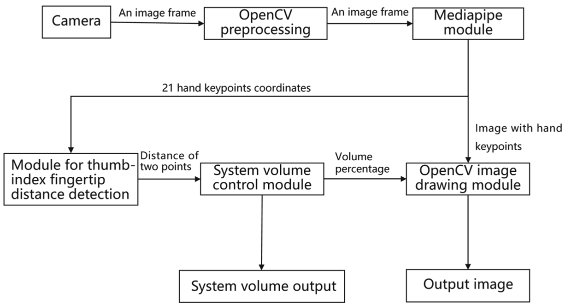
Camera
An image frame
OpenCV preprocessing
An image frame
Mediapipe module
21 hand keypoints coordinates
Image with hand keypoints
Module for thumb-index fingertip distance detection
Distance of two points
System volume control module
Volume percentage
OpenCV image drawing module
System volume output
Output image

Fig. 7. Flowchart for controlling media volume with hand gestures

Experimental Results. Figure 8 shows the actual effect of the gesture-controlled media volume program. When hand is detected, video window labels thumb tip and index finger tip, connects them with a line. Left side shows volume percentage, bottom right corner shows system media volume. Changing distance between thumb tip and index finger tip adjusts volume synchronously for gesture-controlled media volume control.

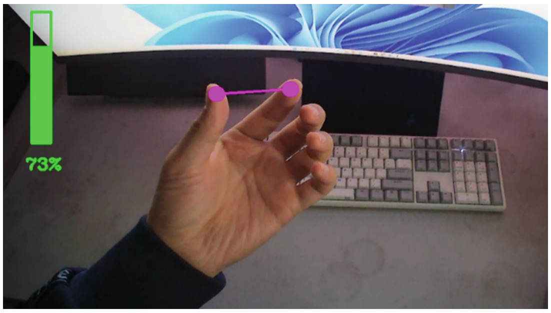

Fig. 8. Image demonstrating media volume controlled by hand gestures

3.5 Test Platform Hardware and Software Composition

Hardware Composition. One computer (Win10 system, AMD Ryzen 9 5950X processor, NVIDIA GeForce RTX 2080 Ti dedicated graphics card, 11GB memory), PICO 4 VR glasses.

Software Composition. PyCharm Community Edition development environment, Python 3.8, PICO Unity Integration SDK.

4 Conclusion

This paper proposes a VR interaction system for emergency rescue in rail transit. It comprises a situation awareness system, dispatch instruction receiver, and VR terminal equipment. The system creates an AR collaboration space for on-site personnel and experts. The VR terminal equipment's human-computer interaction system, using hand recognition, enables functions like virtual mouse, keyboard, and gesture-controlled volume. This study demonstrates the feasibility of gesture recognition in VR input control, expanding its application in rail transit.

Acknowledgements. This study is funded by the National Key Research and Development Program of China (Grant No. 2022YFB4300603) and the National Natural Science Foundation of China (72001021).

References

1. Lv, X., Shi, L., Liao, D.: Application of metaverse in urban rail transit field. Urban Rail Transit **12**, 56–58 (2022). (in Chinese)
2. Qiu, R., He, Z., Ji, F., Xu, W., Song, R.: A review of virtual reality technology based on human-computer interaction in military. Software **42**(6), 123–125 (2021). (in Chinese)
3. Chen, W., et al.: A survey on hand pose estimation with wearable sensors and computer-vision-based methods. Sensors **20**(4), 1074 (2020)
4. Güney, G., Jansen, T.S., Dill, S., et al.: Video-based hand movement analysis of Parkinson patients before and after medication using high-frame-rate videos and mediapipe. Sensors-Basel **22**(20), 7992 (2022)
5. Feng, B., Yang, H., Yang, C.: Research and implementation of simulation platform for track traffic vehicle maintenance based on MR mixed reality technology. J. Shijiazhuang Tiedao Univ. **21**(2), 48–52 (2022). (in Chinese)
6. Bai, H., Yang, S., Feng, X., Song, X., Dong, S.: Study on passenger flow evacuation simulation of rail transit station based on BIM-VR. Urban Mass Transit **34**(5), 60–65 (2021). (in Chinese)
7. Song, D.: Research on the Training System of Shunting Operation in Railway Marshalling Station Based on Virtual Reality Technology. Master's thesis. Beijing Jiaotong University (2020). (in Chinese)
8. Wang, R., Dou, Q., Zhang, Q., Zhou, C.: Gesture recognition based on mediapipe for excavator remote control. J. Civ. Archit. Environ. Eng. **14**(4), 9–16 (2022). (in Chinese)
9. Spurr, A., Dahiya, A., Wang, X., Zhang, X., Hilliges, O.: Self-supervised 3D hand pose estimation from monocular RGB via contrastive learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 11230–11239 (2021)
10. Zhang, F., Dai, G., Peng, X.: A survey of human-computer interaction in virtual reality. Sci. Sin. Inf. **46**(12), 1711–1736 (2016). (in Chinese)
11. Liang, X.: A review of hand pose estimation methods. J. Shanxi Univ. (Natural Science Edition) **45**(3), 631–640 (2022). (in Chinese)
12. Zhang, F., Bazarevsky, V., Vakunov, A., et al.: MediaPipe Hands: On-device Real-time Hand Tracking. arXiv preprint arXiv:2006.10214 (2020)
13. Lugaresi, C., Tang, J., Nash, H., et al.: MediaPipe: A Framework for Building Perception Pipelines. arXiv preprint arXiv:1906.08172 (2019)