Interactable mobile computing framework for augmented reality glasses

Heming Zhang[*a], Ran Lee[b], Jun Wu[c], Kai Yan[d], Yu Pang[e]
[a]Guangdong CAS Cogniser Information Technology Co., Ltd., Guangzhou 511466, Guangdong, China; [b]Mutrics Innovation Technology Co., Ltd., Shenzhen 518000, Guangdong, China; [c]Guangxi CAS Cogniser Technology Development Co., Ltd., Nanning 530028, Guangxi, China; [d]Guangdong Field Vision Technology Co., Ltd., Foshan 440605, Guangdong, China; [e]Chongqing University of Posts and Telecommunications, Chongqing 400065, China

ABSTRACT

Augmented Reality (AR) enhances user interaction with digital content through real-world overlays, finding applications across various fields. AR glasses, serving as an excellent AR platform, have been developed in both optical see-through and video see-through types. Professional video see-through devices and advanced optical see-through devices with vision systems can perform environment recognition and hand detection but are often bulky and heavy for prolonged wear. Conversely, lightweight optical see-through AR glasses, which lack embedded systems and have limited sensors, serve primarily as displays. While they offer the advantage of reduced weight, they lack advanced interaction capabilities. In this research, we utilize an Android mobile phone as the computing unit and present an interactable framework for AR glasses with limited sensors. This framework supports head motion estimation, hand gesture detection and tracking, providing a robust AR experience without the need for high-end hardware. It has been tested on lightweight optical see-through AR glasses only equipped with an Inertial Measurement Unit (IMU) and single camera. Our solution offers a cost-effective and portable approach, enhancing data visualization and virtual object operation.

Keywords: Augmented Reality (AR), AR glasses, user interaction, gesture detection

1. INTRODUCTION

Augmented Reality (AR) is a user-centric technology that enhances engagement with digital content by overlaying it onto the real world. AR applications have been found in various fields, including education, entertainment, industry, and agriculture[1-8].

AR systems can utilize cameras and Time-of-Flight (ToF) sensors to perform environment recognition and interaction. These sensors enable AR devices to map surroundings accurately, detect objects, and provide real-time feedback, enhancing the user experience by creating a seamless blend of digital and physical environments[9-16].

AR glasses as an excellent platform for AR applications, enabling users to interact naturally with augmented content. These glasses come in two primary types: video see through and optical see through[17].

Video see-through AR glasses capture the real-world using cameras and then display the combined digital and real-world view on screens inside the glasses. Video see-through AR glasses tend to have a larger size due to the built-in processing unit and lens structures. In contrast, optical see-through AR glasses use micro-LED projection, and overlay digital content directly onto transparent lenses, enabling users to see both the real and virtual worlds simultaneously. Professional video-see-through AR glasses[18,19] and high-end optical see-through AR glasses[20,21] often include multi-camera vision systems, providing superior environmental recognition and enabling more sophisticated interactions[22-25] with augmented content. However, these advanced capabilities contribute to their larger size or higher hardware costs. Conversely, lightweight optical see-through AR glasses, which lack embedded systems and have limited sensors, serve primarily as displays. While they offer the advantage of reduced weight, they lack advanced interaction capabilities.

In this research, we use a general Android mobile phone as the processing unit and present a mobile framework that can achieve visualization interaction and hand control interaction by using only the AR glasses' built-in IMU (Inertial

[*]zhangheming@cogniser.cn

International Conference on Optics, Electronics, and Communication Engineering (OECE 2024), edited by Yang Yue, Proc. of SPIE Vol. 13395, 1339508 · © 2024 SPIE · 0277-786X · Published under a Creative Commons Attribution CC-BY 3.0 License · doi: 10.1117/12.3049317

Measurement Unit) and single camera, without the need for high-end vision systems or additional hardware[26], making AR technology more accessible and portable.

The framework supports head motion estimation, hand gesture detection and tracking. Head motion estimation allows the system to track the user's head movements, providing a stable and immersive view of the augmented content. Hand gesture detection and tracking enable users to control digital content naturally and intuitively, enhancing the overall user experience. The framework has been tested on an Android mobile phone with lightweight optical see-through AR glasses equipped with a USB Video Class (UVC) camera and an IMU sensor. By leveraging these components, our solution offers a cost-effective and portable approach to AR, enhancing data visualization and virtual object operation.

2. METHOD

In this section, we introduce the hardware used for testing, along with the interaction design, mobile framework, and implementation.

2.1 AR glasses hardware

We utilized a pair of lightweight optical see-through AR glasses as the development hardware[27]. The glasses are equipped with left and right micro displays and lenses, offering a 45-degree field of view (FOV), 6-degree-of-freedom (6-DOF) Inertial Measurement Unit (IMU) and a single camera. The AR glasses can read the DisplayPort output from external devices via USB-C and project it to the lenses. The AR glasses' hardware driver can split a 32:9 image input into left and right images, allowing the user to see a 3D view by outputting left and right side-by-side image frames from mobile devices. The IMU can be accessed as a Human Interface Device (HID), and the camera operates as a USB Video Class (UVC) camera. For this study, we use a general Android mobile phone to communicate with built-in sensors and control the AR glasses' displays to achieve AR functions. The AR glasses and the connection with mobile phone are shown in Figure 1 and their specifications are listed in Table 1.

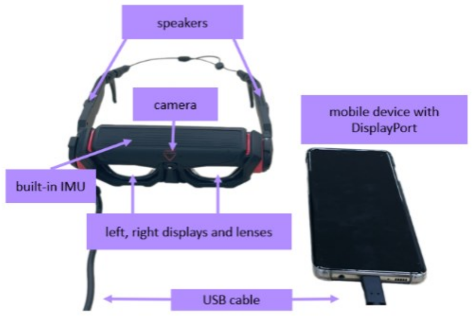

speakers
camera
mobile device with DisplayPort
built-in IMU
left, right displays and lenses
USB cable

Figure 1. Optical see-through AR glasses and connection with mobile phone.

Table 1. Specifications of AR glasses.

Dimension	162 × 51 × 180 mm
Weight	78 g
AR Glasses FOV	45°
Display Resolution	1920 × 1080 per eye
Refresh Rate	60 Hz
Camera Resolution	1280 × 720
Camera FOV	90°
IMU	3-axis accelerometer, 3-axis gyroscope
Port	USB Type C

2.2 Interaction design

2.2.1 Visualization interaction. Visualization interaction is a key feature that leverages the capabilities of AR technology to create a rich and immersive user experience, bridging the gap between the virtual and real world. A fundamental dynamic visualization interaction is used in our study, it allows users to view virtual objects by moving their heads, thereby navigating the AR environment, as shown in Figure 2. The 3D view movements in the AR environment should be synchronized with the user's head movements, creating the illusion that virtual objects exist in the real world.

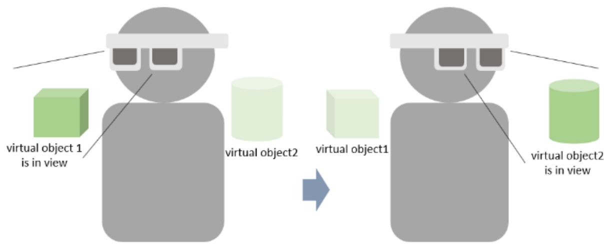


Figure 2. Dynamic visualization interaction.

2.2.2 Hand-based control interaction. To achieve interaction without additional devices, vision systems have been used to capture hand gestures. Multiple cameras can detect depth and obtain 3D and skeleton information[28]. However, such approaches typically require additional and expensive hardware. To address this, we designed a hand gesture control interaction using only a single camera. Our approach utilizes 3 hand gestures: hand open, fist, and pinch, as shown in Figure 3.

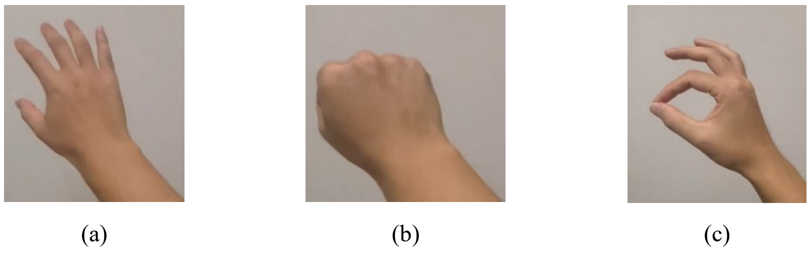


Figure 3. Hand gestures in our approach. (a): hand open gesture; (b): fist gesture; (c): pinch gesture.

Utilizing these simple gestures, our approach provides a cost-effective solution for hand gesture control interaction, eliminating the need for complex hardware setups. By detecting and tracking the gestures and movements, users can perform, move, rotate, and release operations using different gesture combinations, providing an intuitive and natural way to interact with the AR environment. Figures 4 and 5 show rotate and move control interaction using fist gesture and pinch gesture.

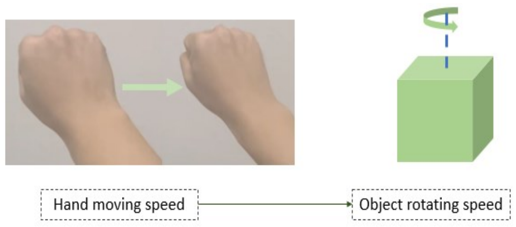


Figure 4. Rotate control using a fist gesture.

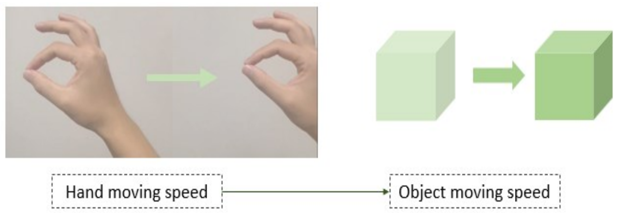
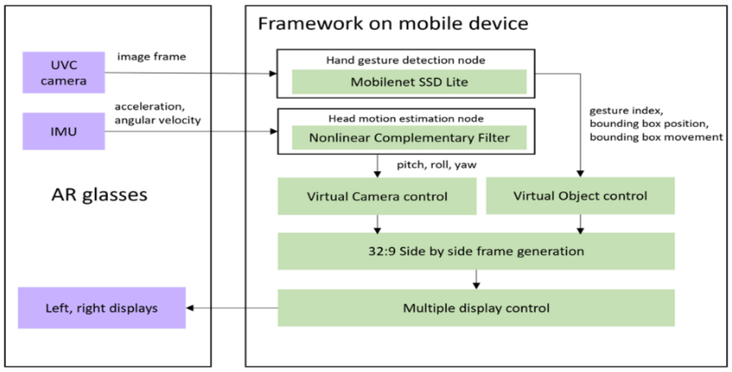


Figure 5. Move control using pinch gesture.

Rotate control allows the user to control the rotation using the first gesture. The gesture's moving speed determines the virtual object's rotating speed: the faster the hand moves, the faster the object rotates. When the hand stops or releases (hand open gesture), the rotation will stop. Move control allows the user to control the movement of virtual objects using the pinch gesture. The gesture's moving speed determines the object's movement speed: the faster the hand moves, the faster the object moves. When the hand stops or releases, the movement will stop.

2.3 Mobile framework design and implementation

The AR glasses serve primarily as a display, requiring an external device for control and rendering. In this study, a general Android mobile phone was used for 3D rendering and control. The 6-DOF IMU data is read at 50Hz, and the camera frames are captured at 30fps. Unity, a 3D engine platform, was employed for development. Within the Unity scene, two cameras were set up to mimic human eye views. By reading the IMU data from the AR glasses, head motion can be estimated and mirrored in the virtual environment. The AR glasses camera captures the user's hand in real time, detecting gestures and movements to control the virtual scene. The system schematic and mobile framework are shown in Figure 6.



AR glasses
UVC camera
IMU
Left, right displays
image frame
acceleration, angular velocity
Framework on mobile device
Hand gesture detection node
Mobilenet SSD Lite
Head motion estimation node
Nonlinear Complementary Filter
gesture index, bounding box position, bounding box movement
pitch, roll, yaw
Virtual Camera control
Virtual Object control
32:9 Side by side frame generation
Multiple display control

Figure 6. Proposed interactable mobile framework for AR glasses with an IMU and single camera.

2.3.1 Mobile framework design and implementation. To implement 2.1.1 head motion estimation, we use a nonlinear complementary filter[29] for IMU data fusion to get head attitude pitch, roll, and yaw. The complementary filter effectively combines data from the gyroscope, which provides accurate angular velocity information but tends to drift over time, with data from the accelerometer, which provides stable orientation information but is susceptible to high-frequency noise and external accelerations. The filter is computationally efficient and well-suited for real-time applications in our AR glasses, ensuring timely and accurate head tracking that is crucial for user interaction and overall system performance.

The head motion estimation flow is illustrated by the following equations. These equations detail the process of combining gyroscope and accelerometer data to compute the head's attitude in terms of pitch, roll, and yaw using a quaternion-based complementary filter.

The estimated gravity vector in the current orientation frame is calculated as:

$$\hat{g}_t = \hat{R}_{t-1}\begin{pmatrix}0\\0\\1\end{pmatrix} \tag{1}$$

Where $\hat{R}_{t-1}$ is the rotation matrix derived from the last quaternion.

The error vector e_t is computed as the cross product of the measured gravity vector from the accelerometer $\tilde{a}_t$ and the estimated gravity vector $\hat{g}_t$:

$$e_t = \tilde{a}_t \times \hat{g}_t \tag{2}$$

The correction term is computed using a proportional-integral (PI) controller:

$$\delta_t = K_p e_t + K_i \sum_i^t e_i \, dt \tag{3}$$

where K_p and K_i are the proportional and integral gains, respectively. This equation ensures that both the current error and the accumulated past errors are considered, providing a balance between immediate correction and long-term stability.

The quaternion differential equation incorporates the angular velocity $\widetilde{\omega}_t$ from the gyroscope and the correction term δ_t, $\otimes$ denotes the quaternion multiplication:

$$\dot{q}_t = \frac{1}{2} q_{t-1} \otimes (\widetilde{\omega}_t + \delta_t) \tag{4}$$

The quaternion q_t is updated by integrating the quaternion differential over the time step dt:

$$q_t = q_{t-1} + \dot{q}_t \, dt. \tag{5}$$

After each update, the quaternion is normalized:

$$q_t = {q_t}/{\| q_t \|} \; . \tag{6}$$

The quaternion can be converted to Euler angles (pitch, roll, and yaw) using the following formulas:

$$pitch_t = \sin^{-1}(q_0 q_2 - q_3 q_1) \tag{7}$$

$$roll_t = \tan^{-1} 2(2(q_0 q_3 + q_1 q_2), 1 - 2({q_1}^2 + {q_2}^2)) \tag{8}$$

$$yaw_t = yaw_{t-1} + \widetilde{\omega}_{zt} dt \tag{9}$$

The yaw angle is calculated by integrating the calibrated z-axis angular velocity $\widetilde{\omega}_{zt}$ to prevent drift. Then, the Euler angles can be used to control the Unity scene cameras.

2.3.2 Hand detection and tracking implementation. To achieve real-time and efficient hand detection and tracking on general mobile devices, the MobileNet-SSD quantized model[30] is used. This model is selected for its balance between accuracy and efficiency, making it suitable for real-time applications on resource-constrained devices such as AR glasses.

To train the model, we followed a systematic process involving data collection and annotation. We labeled a total of 774 images, with 700 images allocated for training and 74 images reserved for evaluation. The dataset includes a diverse range of conditions to ensure robust performance across different scenarios. The images were captured under various lighting conditions, with different backgrounds, at varying distances, and featuring multiple hand postures. Examples of labeled data are shown in Figure 7.

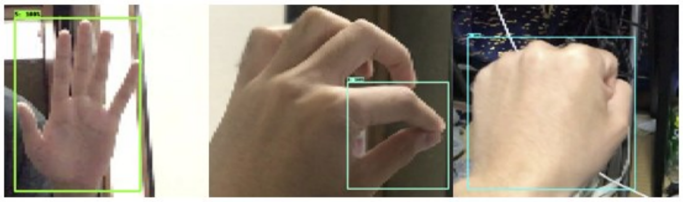


Figure 7. Examples of labeled data used for training the hand detection model.

The bounding box of each detected hand gesture provides feedback on the position within the camera view. By calculating the position derivative between two consecutive camera frames, we can determine the velocity of the gestures. The velocity data is then used to control the rotation and movement of virtual objects, creating an intuitive and responsive interaction.

Assuming the bounding box positions in frame *n*-1 and frame *n* are $\boldsymbol{p_{n-1}}$ and $\boldsymbol{p_n}$ respectively, the bounding velocity can be calculated as follows:

$$\boldsymbol{vbox}_n = \frac{\boldsymbol{p}_n - \boldsymbol{p}_{n-1}}{t} \tag{10}$$

where $\boldsymbol{vbox_n}$ is the speed of the bounding box, $\boldsymbol{p_n}$ and $\boldsymbol{p_{n-1}}$ are the coordinates of the bounding box in frame *n* and frame *n-1*, and *t* is the time interval between the two frames.

By introducing a low-pass (LPF) filter and sensitivity factor *a*, the virtual object's rotation and position are calculated as follows:

$$\boldsymbol{R}_{object} = \alpha_{rotate} \cdot LPF(\boldsymbol{vbox}_n) \tag{11}$$

$$\boldsymbol{P}_{object} = \boldsymbol{P}_{object} + \alpha_{move} \cdot LPF(\boldsymbol{vbox}_n) \tag{12}$$

$$PF(\boldsymbol{vbox}_n) = \beta \cdot \boldsymbol{vbox}_n + (1 - \beta)\boldsymbol{vbox}_{n-1} \tag{13}$$

where $\boldsymbol{R}_{object}$ and $\boldsymbol{P}_{object}$ are Unity game object property object.transform.rotation and object.transform.position, respectively. and are sensitivity factors. To avoid the noise caused by differentiation, a low-pass filter is used to smooth the bounding box's velocity, resulting in more stable control. The low-pass filter for the bounding box velocity $\boldsymbol{vbox}$ is defined using the current n and previous $n-1$ time steps, β is the filter coefficient.

3. RESULTS

3.1 Hand detection training results

We used the collected data introduced in Section 2.2.2 to train the MobileNet-SSD quantized model. Data augmentation such as random horizontal flip and SSD random crop were employed to enhance model robustness.

Figure 8 shows the mAP (Mean Average Precision) at different Intersection over Union (IoU) thresholds during training. The left graph illustrates the mAP@0.50 IoU, indicating the precision of the model at a 50% Intersection over Union threshold, with a final result of 0.990. The right graph illustrates the mAP@0.75 IoU, indicating the precision at a 75% Intersection over Union threshold, with a final result of 0.926. The graphs demonstrate the model's performance improvement over 3000 training steps.

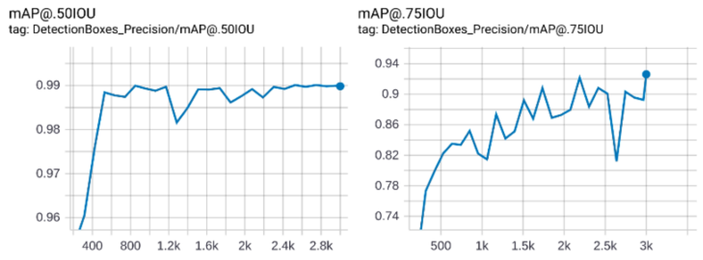


Figure 8. Mean Average Precision (mAP) for hand gesture detection at different IoU thresholds.

Figure 9 shows the training losses over 3000 steps. The left graph illustrates the classification loss, which measures the error in predicting the correct class labels for detected objects, with a final value of 1.858. The right graph illustrates the localization loss, which measures the error in predicting the bounding box coordinates of the detected objects, with a final value of 0.231. Both graphs indicate a decreasing trend, demonstrating that the model is learning effectively and improving its performance over time.

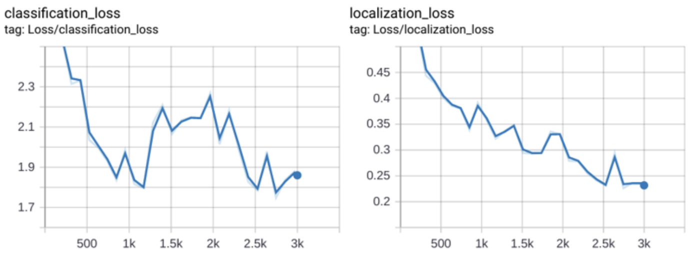


Figure 9. Classification and localization loss during hand gesture detection training.

Figure 10 shows examples of hand gesture detection results. The images on the left in each pair display the detection results, while the images on the right display the ground truth annotations.

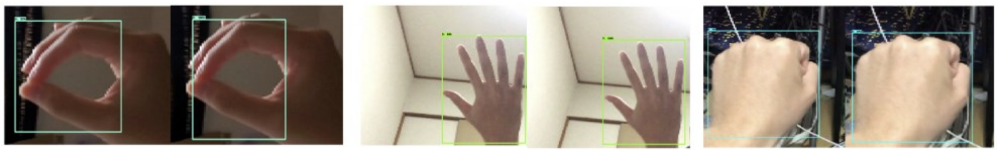
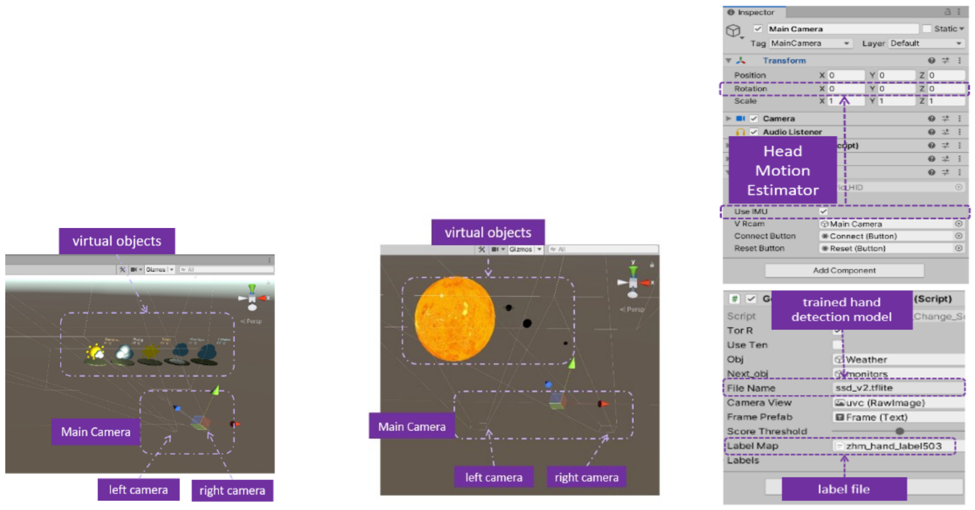


Figure 10. Examples of detection results with trained MobileNet model.

3.2 Demo applications

We developed multiple demo scenes in Unity to evaluate visualization interaction using head motion estimation and hand gesture control integrated with the trained MobileNet model.

Unity demo scene setup and configurations are shown in Figure 11.



```
Main Camera                         Static
Tag MainCamera          Layer Default
Transform
Position   X 0   Y 0   Z 0
Rotation   X 0   Y 0   Z 0
Scale      X 1   Y 1   Z 1
Camera
Audio Listener
Use IMU          ✓
V Rcam           Main Camera
Connect Button   Connect (Button)
Reset Button     Reset (Button)

(Script)
Script           Change_S
Tor R            ✓
Use Ten
Obj              Weather
Next_obj         monitors
File Name        ssd_v2.tflite
Camera View      uvc (RawImage)
Frame Prefab     Frame (Text)
Score Threshold
Label Map        zhm_hand_label503
Labels
```

(a) (b) (c)

Figure 11. Unity scene setup and configurations for visualization and hand control interactions.

(a) 3D weather objects scene, tested for visualization interaction and move control interaction;

(b) Virtual planet scene, tested for rotate control interaction;

(c) Head motion estimation and hand detection model configurations.

Figure 11a shows 3D weather icons were set in the 3D virtual scene. The Main Camera, which includes left and right cameras for stereo vision, has its rotation controlled by the head motion estimator's output. This scene was tested for both visualization interaction and pinch gesture control interaction.

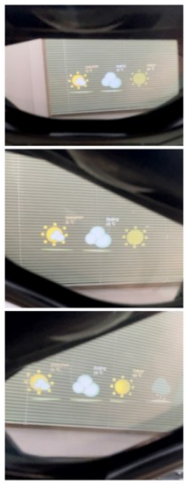
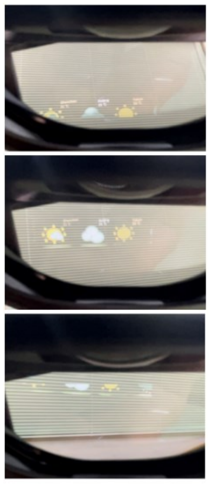
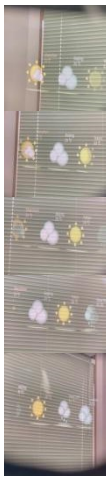
Figure 11b shows the virtual planet scene for testing the first gesture. It displays the layout of virtual objects and the configuration of the Main Camera with its left and right cameras.

Figure 11c shows the configuration settings in the Unity Inspector, including the head motion estimator and the gesture control configurations. The integration of the trained MobileNet model for hand gesture detection and the label file used for detecting gestures are also indicated.

These demo scenes were installed and tested on a Samsung S10 Android phone.

3.2.1 Visualization interaction test. As detailed in section 2.2.1, the head motion estimator processes IMU data from the AR glasses and delivers Euler angles to the Main Camera in Unity application. This setup allows users to move their heads and view virtual objects in the AR environment as if they were suspended in the air.

Figure 12 shows the user's view in the AR glasses when moving the head in different directions. Figure 12 illustrates tilting the head (roll angle), looking up and down (pitch angle), and looking from left to right (yaw angle), respectively. The images from top to bottom represent a time series demonstrating head movement in various directions. The black frame visible at the edges is the frame of the AR glasses.

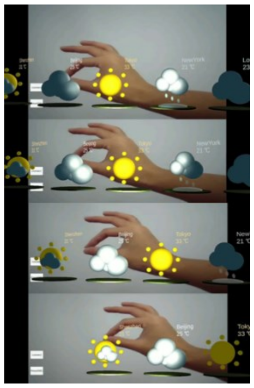
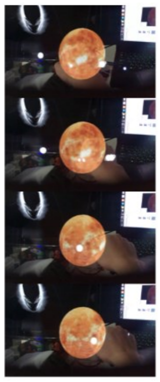
(a) Head tilting (roll angle) (b) Looking from up to down (pitch angle) (c) Turning head from left to right (yaw angle)

Figure 12. Visualization interaction demo.

Figure 12 demonstrates that head motion is accurately estimated and synchronizes the Main Camera's orientation in the virtual environment with the user's head movements. This enables the content displayed in the AR glasses to adjust precisely as the user moves their head in various directions. Consequently, virtual objects appear stable relative to the real world.

3.2.2 Hand gesture interaction test. We utilized the trained hand detection model within the Unity environment to implement the detection and tracking features.

Figure 13a shows the virtual objects' movement when the user uses the pinch gesture to move the objects shown in Figure 11a. Figure 13b shows the virtual objects' rotation when using the pinch gesture to rotate the objects shown in Figure 11b. The images from top to bottom represent a time series demonstrating the progression of these gestures. Figure 13a is a screen recording of the view of the Unity main camera, with the background captured by the AR glasses camera. Figure 13b is recorded from behind the AR glasses lens.

(a) Move control using a pinch gesture, sensitivity factor: 0.09 (b) Rotate control using a fist gesture, sensitivity factor: 0.0005

Figure 13. Hand control interaction demo.

Figure 13 demonstrates that the spatial movement and rotation of virtual objects can be achieved through gesture detection and tracking. By adjusting the sensitivity factor, even small movements within the camera's field of view can be amplified to control the amount of movement and rotation.

4. CONCLUSIONS

The results presented in this study demonstrate a mobile framework for effective hand gesture detection, tracking, and head motion estimation in AR environments using a single camera and IMU. By employing the MobileNet-SSD quantized model, we achieved high accuracy in detecting hand gestures, which facilitated intuitive and responsive interactions with virtual objects. Additionally, the complementary filter used for head motion estimation provided precise tracking of pitch, roll, and yaw movements, enhancing the immersive experience. Our approach proved to be efficient for real-time applications, running smoothly on a mobile device.

The gesture control capabilities extend beyond basic movement and rotation. The system can also interpret gestures to perform actions such as zooming in and out and scaling objects up and down. These additional functionalities enable a richer and more versatile interaction experience in the AR environment, allowing users to manipulate virtual objects in more detailed and nuanced ways.

With the improvement of mobile devices and advancements in AI models, we will implement more sophisticated and intelligent functions on such simple hardware. Additionally, we will focus on applying our approach to real-world AR scenes and applications, such as the remote control of robots and other interactive systems.

ACKNOWLEDGMENTS

This research was funded by a Postdoctoral research project; Nanning Innovation and Entrepreneurship Leading Talents "Yongjiang Plan" Funding Project (2020020) and Industrialization Innovation Team Project of Industrial Technology Research Institute, Foshan Chinese Academy of Sciences (ZK2018009).

REFERENCES

[1] Efstratios, G., Michael, T., Stephanie, B., Athanasios, L., Paul, Z. and George, P., "New cross/augmented reality experiences for the virtual museums of the future. In Digital Heritage. Progress in Cultural Heritage: Documentation, Preservation, and Protection: 7th International Conference, 518-527 (2018).
[2] Capecchi, I., Bernetti, I., Borghini, T., Caporali, A. and Saragosa, C., "Augmented reality and serious game to engage the alpha generation in urban cultural heritage," Journal of Cultural Heritage, 66, 523-535 (2024).
[3] Sara, G., Todde, G. and Caria, M., "Assessment of video see-through smart glasses for augmented reality to support technicians during milking machine maintenance," Sci .Rep., 12, 15729 (2022).
[4] Huuskonen, J. and Oksanen, T., "Augmented reality for supervising multirobot system in agricultural field operation," IFAC-PapersOnLine, 52(30), 367-372 (2019).
[5] Saliha, K., "Role of artificial intelligence and augmented reality in fashion industry from consumer perspective: Sustainability through waste and return mitigation," Engineering Applications of Artificial Intelligence, 133, Part A, 108114 (2024).
[6] Nikoleta, M., Hana, N., Martina, H., Gabriel, F, Vieroslav, M. and Ján, K., "Use of augmented reality in railway transport," Transportation Research Procedia, 77, 253-259 (2024).
[7] Fan, Y., Feng, Z., Mannan, A., Khan, T. U., Shen, C. and Saeed, S., "Estimating tree position, diameter at breast height, and tree height in real-time using a mobile phone with RGB-D SLAM," Remote Sens., 10, 1845 (2018). https://doi.org/10.3390/rs10111845
[8] Santana-Fernández, J., Gómez-Gil, J. and Del-Pozo-San-Cirilo, L., "Design and implementation of a GPS guidance system for agricultural tractors using augmented reality technology," Sensors, 10, 10435-10447 (2010). https://doi.org/10.3390/s101110435
[9] Majed, A. and Manolya, K., "Mobile augmented reality for environmental awareness: A technology acceptance study," ACM Int. Conf. Proceeding Ser., 36-43 (2018).
[10] Alessandro, M., Damiano O., Andrea S., Francesco D. and Federico M., "Snap2cad: 3D indoor environment reconstruction for AR/VR applications using a smartphone device," Computers & Graphics, 100, 116-124 (2021). https://doi.org/10.1016/j.cag.2021.07.014.
[11] Yang, M. D., Chao, C. F., Huang, K. S., Lu, L. Y. and Chen, Y. P., "Image-based 3D scene reconstruction and exploration in augmented reality," Automation in Construction, 33, 48-60 (2013).
[12] Jung, S., Lee, Y.-S., Lee, Y. and Lee, K., "3D reconstruction using 3D registration-based ToF-Stereo fusion," Sensors, 22, 8369 (2022). https://doi.org/10.3390/s22218369
[13] Carbone, M., Domeneghetti, D., Cutolo, F., D'Amato, R., Cigna, E., Parchi, P., Gesi, M., Morelli, L., Ferrari, M.

and Ferrari, V., "Can liquid lenses increase depth of field in head mounted video see-through devices?" Journal of Imaging, 7, 138 (2021).

[14] Itoh, Y., Langlotz, T., Sutton, J. and Plopski, A., "Towards indistinguishable augmented reality: A survey on optical see-through head-mounted displays," ACM Computing Surveys (CSUR), 54(6), 1-36 (2021).

[15] Van, N. S., Le, S. T., Tran, M. K. and Tran, H. M., "Reconstruction of 3D digital heritage objects for VR and AR applications," Journal of Information and Telecommunication, 6(3), 254-269 (2021). https://doi.org/10.1080/24751839.2021.2008133

[16] Fang, W., Zheng, L., Deng, H. and Zhang, H., "Real-time motion tracking for mobile augmented/virtual reality using adaptive visual-inertial fusion," Sensors, 17, 1-22 (2017).

[17] Rolland, J. P., Holloway, R. L. and Fuchs, H., "Comparison of optical and video see-through, head-mounted displays," Telemanipulator and Telepresence Technologies, 2351, 293-307 (1995).

[18] Apple Vision Pro. Available from: https://www.apple.com/shop/buy-vision.

[19] Meta Quest 3. Available from: https://www.meta.com/jp/en/quest/quest-3/

[20] Magic leap2. Available from: https://www.magicleap.com/magic-leap-2

[21] Microsoft. Microsoft HoloLens 2. Available from: https://www.microsoft.com/en-us/hololens.

[22] Rastgoo, R., Kiani, K., Escalera, S. and Sabokrou, M., "Multi-modal zero-shot dynamic hand gesture recognition," Expert Systems with Applications, 247, 123349 (2024).

[23] Radkowski, R. and Stritzke, C., "Interactive hand gesture-based assembly for augmented reality applications," 5th International Conference on Advances in Computer-Human Interactions, 303-308 (2012).

[24] Wen, R., Tay, W. L., Nguyen, B. P., Chng, C. B. and Chui, C. K., "Hand gesture guided robot-assisted surgery based on a direct augmented reality interface," Comput Methods Programs Biomed, Sep, 116(2), 68-80 (2014).

[25] Sebastian, B., David, W. and Annika, R., "A hand-interaction model for augmented reality enhanced human-robot collaboration," CIRP Annals, (2024).

[26] Heinrich, F., Bornemann, K., Polenz, L., Lawonn, K. and Hansen, C., "Clutch & Grasp: Activation gestures and grip styles for device-based interaction in medical spatial augmented reality," International Journal of Human-Computer Studies, 180, 103117 (2023).

[27] Mutrics Aric. Available from: https://www.mutrics.com/products/mutrics-aric

[28] Yang, L., Xu, J., Zhong, L., Zhan, X., Wang, Z., Wu, K. and Lu, C., "POEM: Reconstructing hand in a point embedded multi-view stereo," Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 21108-21117 (2023).

[29] Mahony, R., Tarek, H. and Jean-Michel, P., "Nonlinear complementary filters on the special orthogonal group," IEEE Transactions on Automatic Control, 53(5), 1203-1217 (2008). ff10.1109/TAC.2008.923738ff. ffhal-00488376f

[30] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A. and Chen, L. C., "MobileNetV2: Inverted residuals and linear bottlenecks," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 4510-4520 (2018). doi: 10.1109/CVPR.2018.00474.