



Smart training: Mask R-CNN oriented approach

Mu-Chun Su^a, Jieh-Haur Chen^{b,c,*}, Vidya Trisandini Azzizi^c, Hsiang-Ling Chang^a,
Hsi-Hsien Wei^d

^a Department of Computer Science and Information Engineering, National Central University, Jhongli, Taoyuan 32001, Taiwan

^b Research Center of Smart Construction, National Central University, Jhongli, Taoyuan 32001, Taiwan

^c Department of Civil Engineering, National Central University, Jhongli, Taoyuan 32001, Taiwan

^d Department of Building and Real Estate, Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

ARTICLE INFO

Keywords:

Smart training
Augmented reality
Hand gesture recognition
Finger-pointing analysis
Mask Regions with Convolutional Neural Network (R-CNN)

ABSTRACT

This paper is aimed at the usage of an augmented reality assisted system set up on the smart-glasses for training activities. Literature review leads us to a comparison among related technologies, yielding that Mask Regions with Convolutional Neural Network (R-CNN) oriented approach fits the study needs. The proposed method including (1) pointing gesture capture, (2) finger-pointing analysis, and (3) virtual tool positioning and rotation angle are developed. Results show that the recognition of object detection is 95.5%, the Kappa value of recognition of gesture detection is 0.93, and the average time for detecting pointing gesture is 0.26 seconds. Furthermore, even under different lighting, such as indoor and outdoor, the pointing analysis accuracy is up to 79%. The error between the analysis angle and the actual angle is only 1.32 degrees. The results proved that the system is well suited to present the effect of augmented reality, making it applicable for real world usage.

1. Introduction

Compared to traditional screens, smart glasses combined with augmented reality technology can present a more realistic picture, making it widely useful in entertainment, medical, military, education, and other major fields. Augmented reality has become a technology trend in human life, and therefore, changing human's lifestyle with it. Augmented Reality (AR) is a view of a real-world environment that is modified by a computer. It is a subset of Virtual Reality (VR) but differs from VR in that it offers a greater sense of realism to its users. Augmented reality is a technology that combines virtual world with reality. Through a camera or a head-mounted device, virtual information such as text, patterns, 3D models and other objects are added to the real world. The information does not need to be presented on a separate display. Instead, it is directly presented in front of the user's eyes through the penetrating vision system and combined with the user's senses. Computer vision renders 3D virtual objects from the same viewpoint from which the images of the real scene are being taken by tracking cameras (Carmigniani et al., 2011). In AR, the environment is real, but extended with information and imagery from the system, thus allowing it to bridge the gap between what is real and what is virtual coherently. The usage of AR is enormous. Regarding city tourism,

Manchester Metropolitan University (MMU), Dublin Institute of Technology (DIT) and the Dublin City Council arranged a project called Dublin AR project. This project aims to provide a platform to superimpose tourism relevant information, reconstruct and revive stories of the past, assisting the tourist in creating an emotional experience of the intangible product. In education sector, AR is mostly applied in the classroom-based learning within the scope of scientific subjects such as mathematics, physics, biology, and chemistry through augmented books and student guides (Lee, 2012). In industrial and technical military operations, AR is hugely helpful for the training process due to complexity of maintenance and assembly tasks. AR allows a highly efficient and cost-effective method of training, especially in procedural skills. Procedural skills are the ability to follow repeated a set of actions step-by-step in order to achieve a specified goal.

The research proposes a set of AR assistance system using Mask Regions with Convolutional Neural Network Features (R-CNN) approach. The system captures the pointing gesture from the image and analyzes it. At the same time, it uses the calibration of objects in the image to analyze the rotation angle of the virtual auxiliary tool. The above three technologies can be used to fully present the use of the system. The system can achieve the following sub-goals: To present the related information of objects in augmented reality more realistically in

* Corresponding author.

E-mail addresses: muchun@csie.ncu.edu.tw (M.-C. Su), jhchen@ncu.edu.tw (J.-H. Chen), hhwei@polyu.edu.hk (H.-H. Wei).

<https://doi.org/10.1016/j.eswa.2021.115595>

Received 16 December 2020; Received in revised form 4 May 2021; Accepted 10 July 2021

Available online 17 July 2021

0957-4174/© 2021 The Author(s). Published by Elsevier Ltd. This is an open access article under the CC BY license (<http://creativecommons.org/licenses/by/4.0/>).

front of users; to build the object detection function on the device, including the creation of object data set and training network; to show information corresponding to the pointing gesture; and to adjust the display angle of virtual auxiliary tool in accordance to the user's lens rotation.

2. AR and mask R-CNN applications

Convolution neural network (CNN) is a class of deep neural networks, also known as shift invariant or space invariant artificial neural networks. It is mainly divided into two parts. The first part is input part, consisting of unsupervised multiple convolution layers and pooling layers. The second part consists of supervised fully connected layer. Convolution layer is the core part of the network, creating different representations of the learning dataset, starting from more general ones at the first larger layers, becoming more specific at the deeper layers (Alhaija, Mustikovela, Mescheder, Geiger, & Rother, 2018). It is able to extract features such as edges, lines, and corners, and the more complex features can be extracted over multiple layers. Therefore, based on smart glasses as a human-machine interface, it provides a humanized way to display the object name that the operator wants to know by pointing gestures. It can specifically be used for the training of machine operators and device maintenance personnel. In the near future, machines can replace manpower, saving time and cost of training personnel for the industry, making it act as the new milestone opening for the industry. When the user wears the smart glasses, the pointing gesture can show the relevant information of the finger-pointing at the object and the corresponding virtual tool. Through the Mask Regions with Convolutional Neural Network (Mask R-CNN) network training, a system object detection function can be created to mark the object data set. The image uses the Labelme tool to mark each object in order to create a training data set. The car engine room is used as an example of the device. The marked objects included the air filter, auxiliary water tank, battery, brake oil pot, engine, fuse box, oil cap, and wiper water bottle, including images of object rotation, distance and displacement. After many experiments, it is concluded that as long as the number of marked object data sets reaches about 110 pieces, and it contains images of objects with different rotation angles and different distances, it can produce consistent results. The data set is trained through the Mask R-CNN. In the application of object detection system, an image act as an input to the trained network model, and the object name, position and mask information become the output. The pointing analysis algorithm is divided into two steps. The first part is the acquisition of gestures. Mask R-CNN is used to train single finger pointing gestures. The resulting block image is then extracted by the skin color algorithm according to the current situation. The second part is the pointing analysis. According to the extracted hand, the fingertip points and finger centroids are detected and corresponded. The picture of the main bracket to the hand is the pointing of the finger.

Practitioners in both academic and industries usually expect 3 main functions for AR technologies: simultaneous existence of reality and virtuality, interaction, and 3 dimensional operation. AR technologies have been applied widely. We can see AR application on tourism, learning and teaching development, military, car repair and maintenance, and so on (Gharaibeh, Gharaibeh, Khan, Abu-ain, & Alqudah, 2021; Jang, Ko, Shin, & Han, 2021; Lee & Rhee, 2008; Ullo, Piedimonte, Leccese, & De Francesco, 2019; Yin, Jung, Dieck, & Lee, 2021). To achieve these applications, there are 6 typical techniques named R-CNN, Fast R-CNN, Faster R-CNN, Mask R-CNN, You Only Look Once (YOLO), and Single Shot multi-box Detector (SSD). Table 1 demonstrates the comparison among the techniques. For better accuracy, R-CNN and its applications outperform the other techniques (Bello, Mohamed, & Talib, 2021; Iqbal, Basit, Ali, Babar, & Ullah, 2021; Li, Qu, Wang, & Liu, 2021; Ren, He, Girshick, & Sun, 2016; Xiang, Seeling, & Fitzek, 2021). Mask R-CNN has been introduced recently to improve Faster R-CNN by adding Region of Interest (ROI) align to carry out object detection and semantic

Table 1

Comparison among major object detection techniques.

Techniques	Mechanism	Efficiency	Accuracy
R-CNN Faster R-CNN Mask R-CNN	Region Proposal	Low	High
YOLO SSD	Regression	High	Low

segmentation simultaneously.

3. AR assistance system using Mask R-CNN approach

The system architecture shown in Fig. 1 is divided into two parts, Client and Server. The client uses the Moverio BT-300 smart glasses developed by EPSON, and the server developed using Unity. It is the core algorithm which performs object detection, pointing analysis, and virtual tool positioning and rotation angle analysis operations. The system of smart glasses is divided into two scenarios, scenario 1 presents all objects, and scenario 2 presents selected objects. At the beginning, the information of all the objects on the device is presented so that the user can know which objects are available. When the user is particularly interested in an object, the user points his finger at the object, that is, the situation is converted to 'point to the object', and all objects are no longer displayed. The device only displays information of the objects that was pointed at.

3.1. Pointing gesture capture

If the image gesture is captured directly by skin color detection, it is susceptible to light and background color (Randive, Mali, & Lokhande, 2012). The effect is not obvious, so this paper proposes a new gesture capture algorithm based on Mask R-CNN and skin color detection. 400 gesture images (including the hands of ten people) and shooting pointing gesture images with different rotation angles and different distances were marked in the Mask R-CNN training pointing gesture data set. In actual application, an image act as an input to obtain the position and mask information of the gesture, so the gesture can be captured according to the current light of the image to reduce the missing skin color information. The image was first converted to the HSL color space (Kolkur, Kalbande, Shimpi, Bapat, & Jatakia, 2017). H stands for Hue which represents the attribute of color. S stands for Saturation is the saturation, the larger the value, the more intense the image color is. L stand for Lightness/brightness, where the larger the value, the brighter the image color is. Equations (1)–(5) are the conversion Equation of the color space used to transfer the image from RGB color space to HSL. RGB represents the red, green, and blue values of the pixel, V_{\max} is the maximum value in red, green and blue, V_{\min} is the minimum value in red, green (Köpüklü, Gunduz, Kose, & Rigoll, 2019; Loesdau, Chabrier, & Gabillon, 2014; Rafique, Jalal, & Kim, 2020; Xie & He, 2016).

$$V_{\max} = \max(R, G, B) \quad (1)$$

$$V_{\min} = \min(R, G, B) \quad (2)$$

$$L = \frac{V_{\max} + V_{\min}}{2} \quad (3)$$

$$S = \begin{cases} \frac{V_{\max} - V_{\min}}{V_{\max} + V_{\min}} & \text{if } L < 0.5 \\ \frac{V_{\max} - V_{\min}}{2 - (V_{\max} + V_{\min})} & \text{if } L \geq 0.5 \end{cases} \quad (4)$$

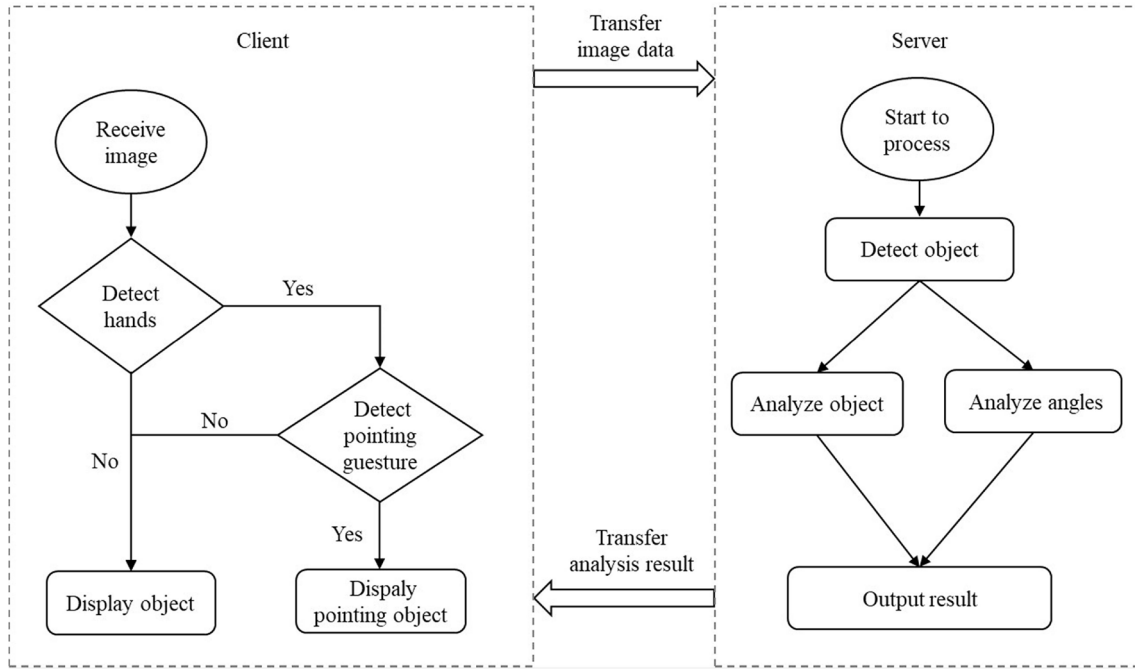


Fig. 1. System architecture.

$$H = \begin{cases} 60 \times \frac{(G - B)}{S} \text{ if } V_{\max} = R \\ 120 + 60 \times \frac{(B - R)}{S} \text{ if } V_{\max} = G \\ 240 + 60 \times \frac{(R - D)}{S} \text{ if } V_{\max} = B \end{cases} \quad (5)$$

The gestures has a high separation between hue and saturation, so HSL color space was used instead of other image color spaces (Ehkan, Siew, Zakaria, Warip, & Ilyas, 2020; Nishad, 2013; Saravanan, Yamuna, & Nandhini, 2016). Finger-pointing gesture were trained using Mask R-CNN, based on the detected pointing gesture block and mask, where the image pointing gesture block were used subsequently. In Eq. (6), Pixel represents pixel; Not skin represents non-color pixel; Skin represents skin color; Pixel_H represents hue value of the pixel; Mean_H represents average hue value; Std_H represents the hue standard deviation value. Mask R-CNN's mask removes blocks with large color differences, and the remaining mask blocks are defined as skin color masks. The average and standard deviation of the three HSL channels are calculated as hand skin color information.

$$\text{Pixel} \in \begin{cases} \text{NotSkin} & \text{if } |\text{Pixel}_H - \text{Mean}_H| > \text{Std}_H \\ \text{Skin} & \text{else} \end{cases} \quad (6)$$

Using Eq. (7) where SkinMean_H is the average hue value of the skin mask and SkinStd_H is the standard deviation of the hue value of the skin mask, the image gesture block are intercepted by Mask R-CNN and the area where the color difference of the skin color mask is too large are removed. Using Eq. (8), where Not Hand represents non-hand portion of the image; Hands represents the hand portion of the image; Pixels represent saturation value of the pixel; Skinmeans represents average saturation value of skin mask pixels; SkinStd_s represents the standard deviation of the saturation skin color mask. Using similar Pixel Equation in Eq. (6), saturation removes areas with large differences, but retains the areas detected by the skin color algorithm. The purpose is to preserve the information of the hand.

$$\text{Pixel} \in \begin{cases} \text{NotSkin} & \text{if } |\text{Pixel}_H - \text{Mean}_H| > \text{Std}_H \\ \text{Skin} & \text{else} \end{cases} \quad (7)$$

$$\text{Pixel} \in \begin{cases} \text{NotHand} & \text{if } |\text{Pixel}_s - \text{Mean}_s| > \text{Std}_s \\ \text{Hand} & \text{else} \end{cases} \quad (8)$$

If the hue of the image and the skin mask are too different, they indicate that there are non-hand objects in the image. Using Eq. (9) where Object stands for object and the definitions of Pixel, Hand, Pixels, SkinMeans, Skinstds are still the same as Eq. (8), image noise is further reduced. Eq. (10) were used to remove the extreme saturation area of the image block, the purpose of the equation is to filter the background at hand. Finally, the filled maximum contour is taken as the result of gesture capture.

$$\text{Pixel} \in \begin{cases} \text{Object} & \text{if } |\text{Pixel}_s - \text{SkinMean}_s| > \text{Std}_s \\ \text{Hand} & \text{else} \end{cases} \quad (9)$$

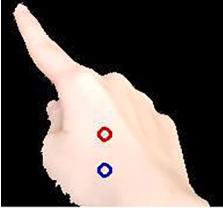
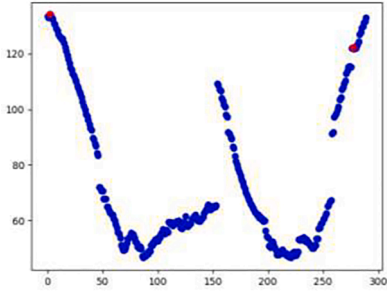
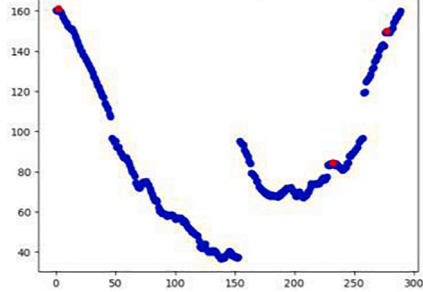
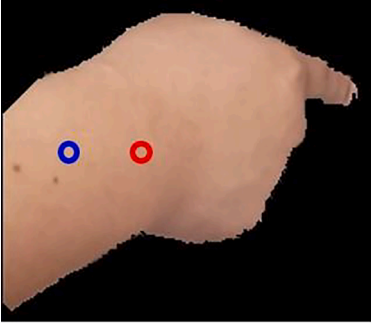
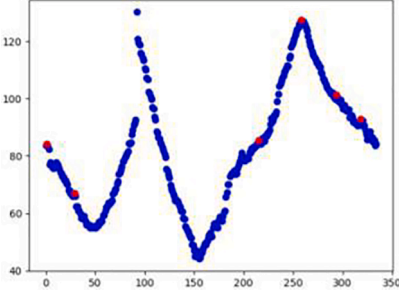
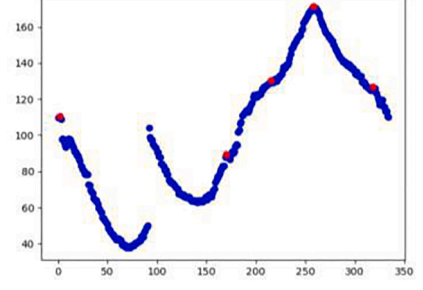
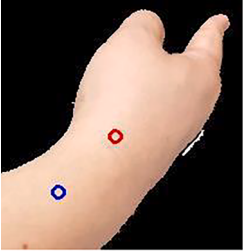
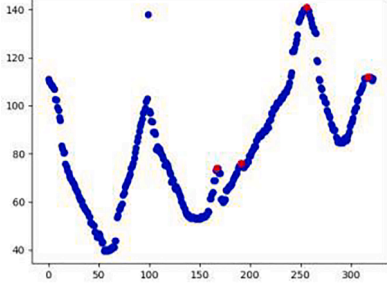
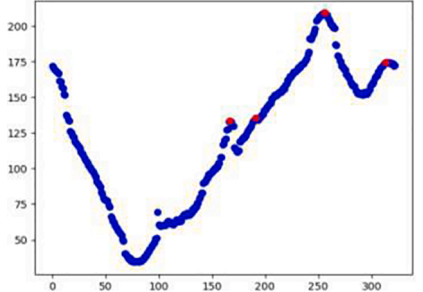
$$\text{Pixel} \in \begin{cases} \text{Background} & \text{if } |\text{Pixel}_s - \text{Mean}_s| > \text{Std}_s \times 1.5 \\ \text{Hand} & \text{else} \end{cases} \quad (10)$$

Finger-Pointing Analysis

First, the result of gesture capture is defined as hand's distance curve feature reference. The hand's center of mass were used to pinpoint the palm of the hand, and the distance from all contour points of the hand to the palm were calculated to generate a distance map. To find the direction accurately, Eqs. (11)–(14) were used to shift the palm of the hand (Ren, Yuan, Meng, & Zhang, 2013). (HandCenterX , HandCenterY) represents the shifted coordinate palm from (OriCenterX , OriCenterY). LeftX is left border contour value of coordinate X, RightX is right border contour value of coordinate X, TopY is upper border contour value of coordinate Y, and Bottom Y is lower border contour value of coordinate. If the left border contains skin color pixels greater than one-third of the outline height, the left border is regarded as the wrist area, and the palm of the hand is shifted to the left using Eq. (11). Similarly, if the right border is the wrist area, Eq. (12) displace the palm to the right; if the upper boundary contains skin-tone pixels greater than one-third of the outline width, Eq. (13) shifts the palm up, and if the lower boundary is the wrist area, Eq. (14) shifts the palm down. The expected fingertip point is farther from the palm of the hand, thereby increasing the weight of the fingertip shown in the upper part of Table 2.

$$\text{HandCenterX} = \text{OriCenterX} - (\text{OriCenterX} - \text{LeftX})/2 \quad (11)$$

Table 2
Original and Adjusted Palm Distance Map.

The red dot is the original palm position (center of mass), and the blue dot is the adjusted palm position	Contour and red dot distance map (X and Y-axis: distance in millimeter)	Contour and red dot distance map (X and Y-axis: distance in millimeter)
		
		
		

$$\text{HandCenterX} = \text{OriCenterX} + (\text{RightX} - \text{OriCenterX})/2 \quad (12)$$

$$\text{HandCenterY} = \text{OriCenterY} - (\text{OriCenterY} - \text{TopY})/2 \quad (13)$$

$$\text{HandCenterY} = \text{OriCenterY} + (\text{BottomY} - \text{OriCenterY})/2 \quad (14)$$

The fingertip point can only be pinpointed after determining the palm point of the hand, using its shape and high curvature. The palm and distance map were first needed to be taken out (Lee & Lee, 2011). All the peak contour points of the image are marked as fingertip candidate points, and each block of fingertip's candidate is framed. Finger's characteristics are elongated and the wrist is a blocky plane, therefore, the calculation can be performed. First, the blocky plane need to be removed. For each fingertip candidate block, the color border number of each row and each column were calculated. The rows and columns that contain two black and white border are regarded as finger regions. Second, the white pixels of the row and column for the row and column containing 2 junctions were counted. Since the fingers are elongated, the number of pixels in each row or column is consistent, so the smoothest line segment is needed to avoid misjudgment due to the large area of the block selected by the fingertip shown in the middle part of Table 2. Eq. (15) is used to remove the block parts where Preserve represents a point on the line, Remove reserved for the rounding point, LineMean is the average point of the segment, neStd is the standard line point of difference. Finally, the frame with the lowest slope in the gentle area is selected as the final fingertip shown in the lower part of Table 2.

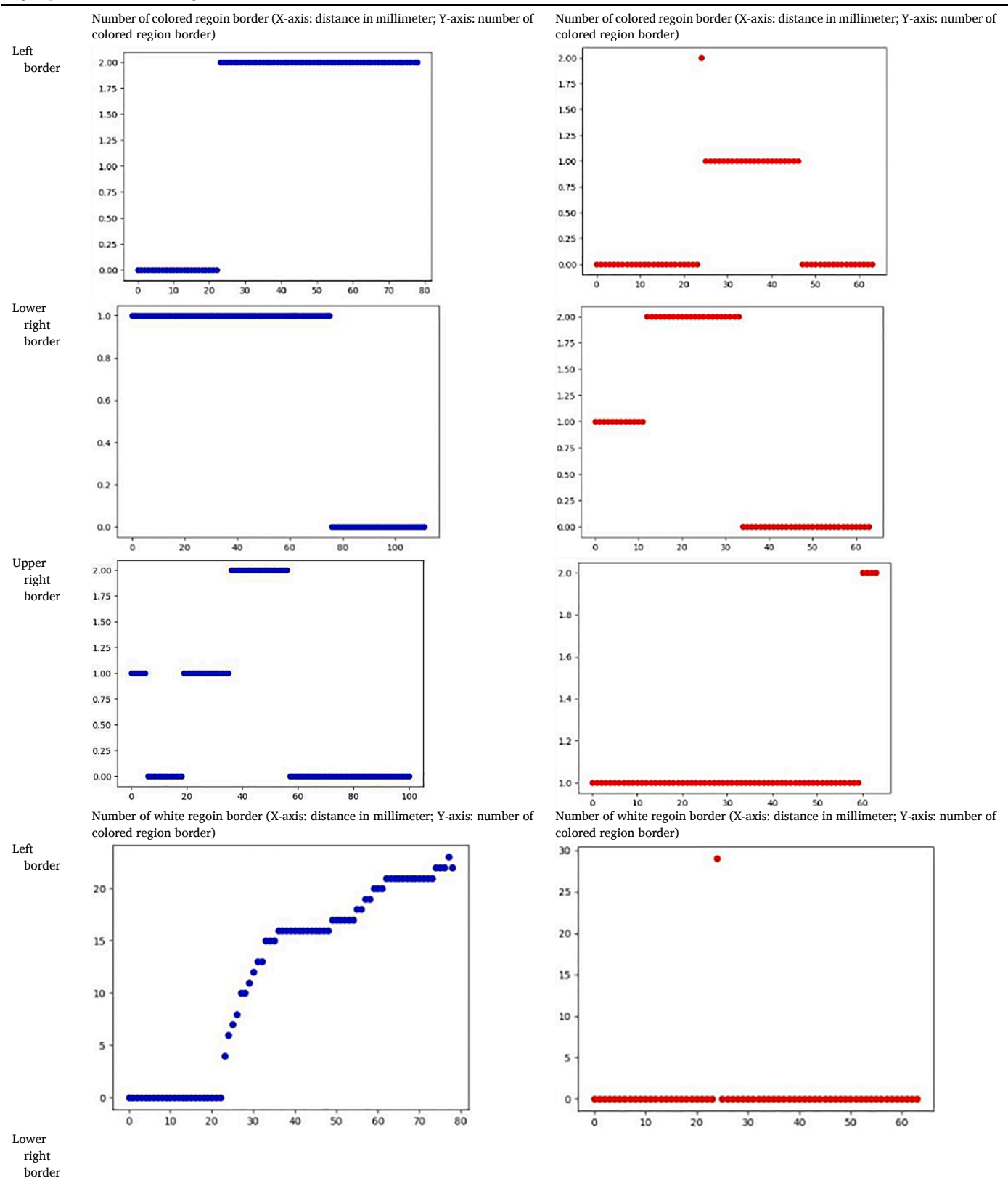
$$\text{Point} \in \begin{cases} \text{Preserve} & \text{if } |\text{Point} - \text{LineMean}| \leq \text{LineStd} \times 0.8 \\ \text{Remove} & \text{else} \end{cases} \quad (15)$$

After finding the fingertip point, then take the two points closer to the palm of the hand near the peak of the fingertip point in the figure. The triangle block formed by the fingertip point and the two left and right contour points, finally, the center point is used as the finger centroid. Finally, the result of the gesture capture is thinned to obtain the main bracket diagram of the hand. Using two points closest to the centroid and fingertip points of the finger, and form the pointing vector according to the direction of the main bracket diagram shown in Table 3.

3.2. Virtual tool positioning and rotation angle

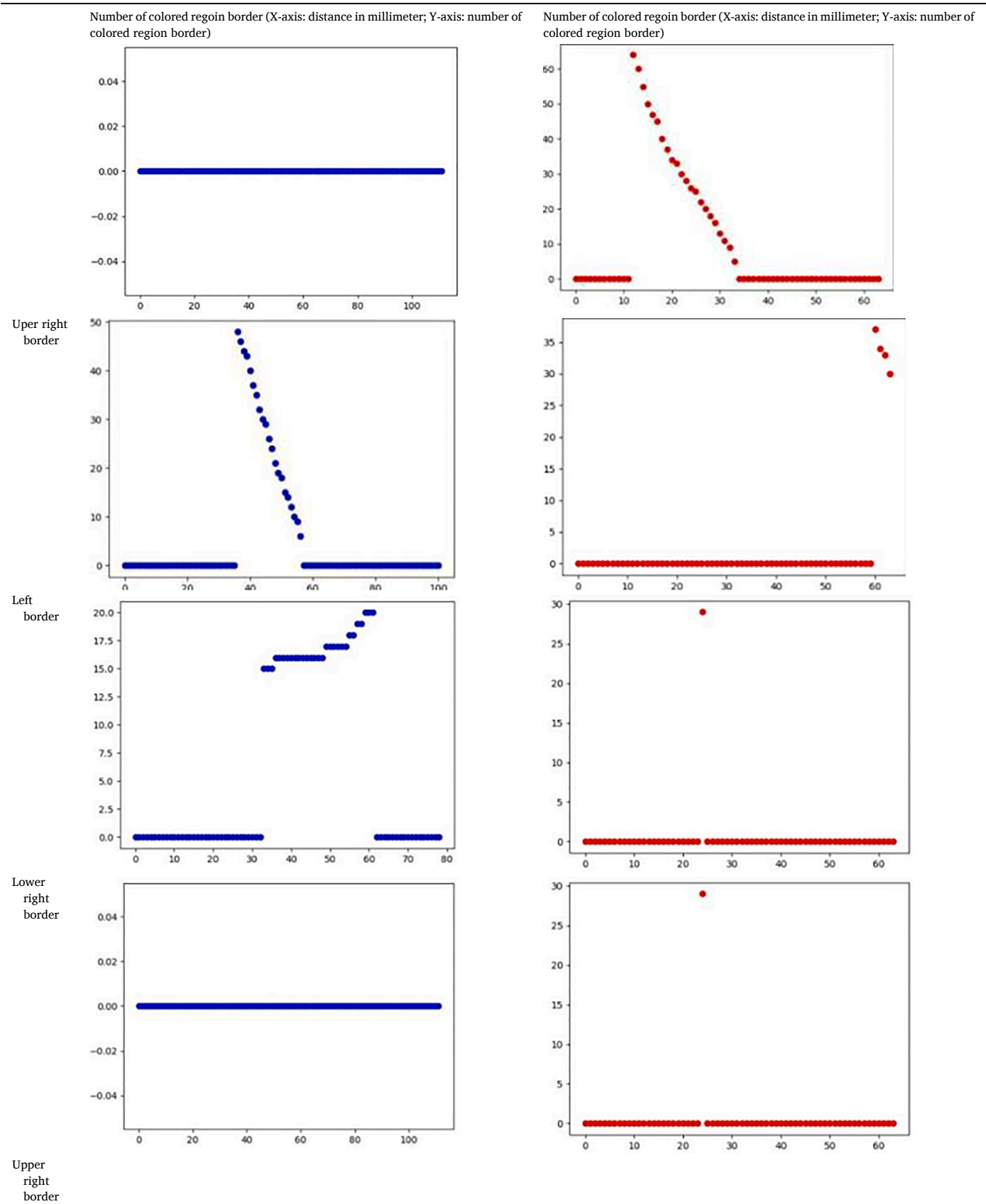
The system uses smart glasses where the lens rotate with the user's head instead of the fixed camera. If the operation positioning and steering angle of virtual tools are known, the effects of augmented reality virtual tools can be presented more accurately. Taking the car engine room as an example of the installation, virtual tools were used on the air filter, engine and battery objects, and also use the Labelme tool to mark the production of various screw parts on the air filter, engine and battery objects. Each type of screw parts required for marking the training data set contains images of object rotation, distance and displacement, and then the data set is trained through the Mask R-CNN network to complete the part of the object screw parts detection (Deng et al., 2017). Two-layer network were utilized for the virtual tool

Table 3
Fingertip number of colored regoin border.



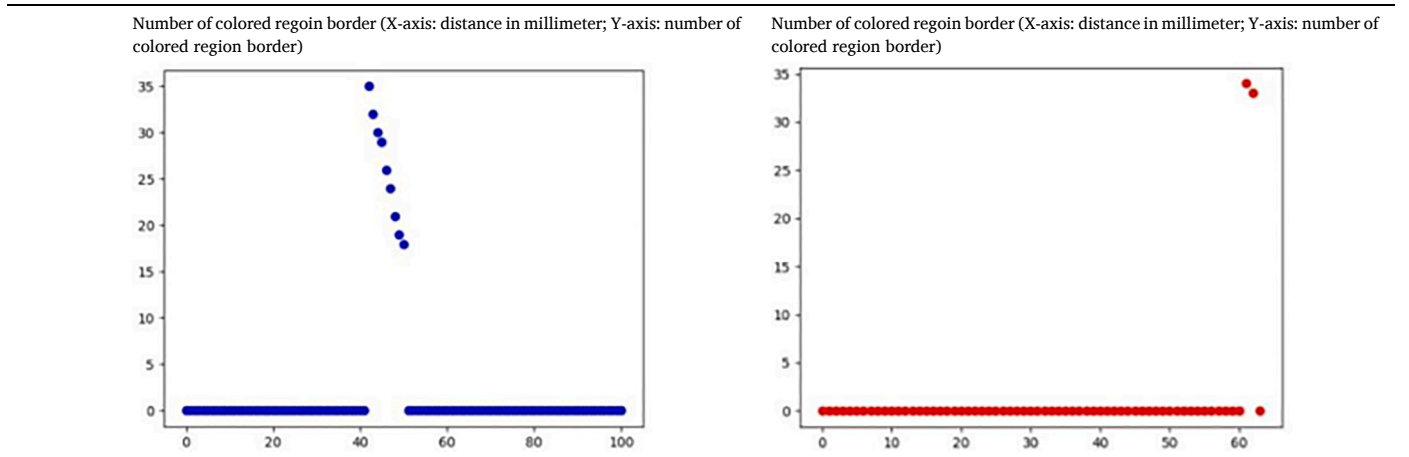
(continued on next page)

Table 3 (continued)



(continued on next page)

Table 3 (continued)



positioning flowchart as shown. The image were inputted to the first-layer network for object detection. In order to reduce the computing time, only if an object used by the virtual tool is detected, the image went on to the second layer network to detect the position of the screw parts, which completes the positioning of the virtual tool. Objects calibration were used to produce the function of analyzing the steering of virtual tools. The rectangular calibration object in the device were selected, and the image inputted into the trained model through Mask R-CNN training to obtain the position of the calibration object on the image. After taking out the box image of the Mask R-CNN calibration object, the image converted to a grayscale image, and then it was binarized in order to find the largest contour of the minimum bounding rectangle for object calibration purposes. Afterwards, the rectangle is rotated to provide information for perspective analysis. The minimum bounding rectangle uses the algorithm of OpenCV. The information included the center point, width, height and angle. The angle is calculated by scanning the image from bottom to top by a horizontal line. When the horizontal line hits the outer rectangle for the vertex, the first side it encounters when it rotates counterclockwise is set as Width, and the other side is set as Height, and the angle when it rotated counterclockwise is the circumscribed rectangle angle (Dongare, 2014). Based on this information, the angle and rotation direction of the augmented reality virtual tool can be derived. Eq. (16) is the derived Equation for the rotation direction of the virtual tool. Turn represents the direction of rotation, Width is the width of the calibration object, and Height is the length of the calibration object which was considered to be greater than the length of the calibration object. If the width of the object is greater than the length, and it is considered clockwise rotation. Otherwise, the object is considered counterclockwise rotation.

$$\text{Turn} = \begin{cases} \text{Clockwise, if Width} < \text{Height} \\ \text{Counterclockwise, if Width} > \text{Height} \end{cases} \quad (16)$$

The derivation Equation of the virtual tool rotation angle can be seen in Eq. (17), where Tool Angle represents the tool rotation angle. θ is the angle of the calibration object and Turn is the direction of rotation. When the rotation direction is clockwise, the tool rotation angle is 90 degrees minus the circumscribed rectangle angle. When the rotation direction is counterclockwise, the tool rotation angle is the circumscribed rectangle angle.

$$\text{ToolAngle} = \begin{cases} 90 - \theta, \text{ if Turn} = \text{Clockwise} \\ \theta, \text{ if Turn} = \text{Counterclockwise} \end{cases} \quad (17)$$

4. Evaluation and implementation

4.1. Engine room object detection

The first experiment uses the engine room as the experimental device to detect ten objects, namely the air filter, the auxiliary water tank, the battery, the brake oil can, the engine, the fuse box, the oil cap, the oil dipstick, the steering oil can, and the wiper water tank shown in Fig. 2. Each object in the Mask R-CNN training data set contains about 100 images. Since the training data set is built using Samsung S8 mobile phone lens images, in order to determine whether the object detection accuracy is affected by the camera lens, this experiment uses two shooting lenses to shoot, respectively, Samsung S8 mobile phone 1200 Megapixel camera lens and 5-megapixel camera lens built in BT-300 smart glasses. Using two lenses, a video of about 20 s was recorded. The video contains objects rotated 90 degrees clockwise, 90 degrees counterclockwise, and objects with different distances from the lens. The video serves as a template for the experiment.

In each video, an image is captured every 5 frames for object detection. Once an object is detected, the detected object is marked in this frame of image. The detection rate is calculated as Eq. (18). Among them, Accuracy represents the correct rate of object detection, TN is the total number of frames taken out of each video, and CN is the number of frames that correctly detected the object. The recognition rate is used as the evaluation standard for object detection.

$$\text{Accuracy} = \frac{\text{CN}}{\text{TN}} \quad (18)$$

In the object training data set of Mask R-CNN, there are about 100 images for each object (including images with different rotation angles and different distances). Each video in the experiment can capture about 100 images for testing. The overall test recognition rate is 80% shown in Table 4.

It can be seen that the recognition rate of most objects is quite good, some even reaching 100% accuracy rate, but the recognition rate of the oil dipstick is very bad. After inspection, it is found that the training data set is not marked enough and the angles taken by the two lenses are not in the training data set, which means that the training data of the oil dipstick does not cover all angles. Therefore, ten test images were randomly picked from each object's training data set, and retrained using the Mask R-CNN network. The overall test accuracy has been increased significantly. Thus, only a small amount of training data needs to be added to greatly improve the accuracy of object detection shown in Table 5.



Fig. 2. Components in car engine room for detection.

4.2. Motherboard object detection

In order to prove that the system can be used in different situations, second object detection experiment was conducted. This experiment uses the motherboard as the experimental device. Six objects on the motherboard are detected, each with 8 Pin power sockets, CPU slot, memory slot, 24 Pin power slot, expansion card slot and hard disk slot shown in Fig. 3. Each object in the Mask R-CNN training data set contains about 300 images, training data set creation directly used the images taken by the built-in 5-megapixel camera lens of BT-300 smart

glasses. In the motherboard object training data set of Mask R-CNN, there are about 300 images for each object, including object images with different rotation angles and different distances. Each video in the experiment can capture about 150 images for testing. The overall test recognition rate is 95% shown in Table 6.

4.3. Reaction time for gesture detection

This experiment was carried out indoors under fluorescent lights. A poster car's engine room were printed and placed on the desktop, and

Table 4
Engine Room Object Detection Experiment Results.

Object	Training Data Set Recognition Rate (%)	Samsung S8 Recognition Rate (%)	BT-300 Smart Glasses Recognition Rate (%)
Air Filter	100	93	97
Auxiliary Water Tank	100	100	67
Battery	100	100	100
Brake Oil Can	100	86	78
Engine	99	99	99
Fuse Box	100	71	48
Oil Cap	90	77	72
Dipstick	86	1	1
Oil Tank	100	100	100
Wiper Reservoir	100	100	100
Average	98	83	76

Table 5
Object Detection Result of Retraining.

Object	Training Data Set Recognition Rate (%)	Samsung S8 Recognition Rate (%)	BT-300 Smart Glasses Recognition Rate (%)
Air Filter	100	81	95
Auxiliary Water Tank	100	100	100
Battery	100	100	100
Brake Oil Can	100	100	100
Engine	98	100	100
Fuse Box	100	100	96
Oil Cap	91	70	98
Dipstick	89	87	79
Oil Tank	100	100	100
Wiper Reservoir	100	100	100
Average	98	94	97



Fig. 3. Motherboard sample.

experiment's subject were asked to point the object in the engine room to shoot the experiment template. At the moment of the experiment, the pointing gesture data set of this system has 300 images, including the hands of eight people shooting various pointing gesture images with different rotation angles and different distances, and the detection accuracy of the training data set is 100%. Since the training data set was built using images taken by the Samsung S8 mobile phone lens, in order to determine whether the gesture detection is affected by the camera lens, this experiment uses two shooting lenses to shoot, namely the

Table 6
Motherboard Object Detection Experiment Results.

Object	Training Data Set Recognition Rate (%)	BT-300 Smart Glasses Recognition Rate (%)
8 Pin Power Socket	96	93
24 Pin Power Socket	99	88
CPU Socket	100	100
Memory Slot	100	100
Expansion Card Slot	100	99
Hard Drive Slot	100	93
Average	99	95

Samsung S8 mobile phone lens and the BT-300 smart glasses. In order to mimic the built-in camera lens in the smart glasses, the test subject were asked to take pictures using the mobile phone in front of their eyes, thus simulating the situation of smart glasses.

The subjects were asked to stand in front of the device in six positions: near left, near middle, near right, far left, far, and far right, where the distance between the device and the subject is about 10 cm for the 'near' positions and 50 cm for the 'far' positions. The subject points to three objects on the device in each position: auxiliary water tank, engine, and battery. The objects are located on the left, middle and right of the device. Each subject stands at each position and points to each object and recorded a pointing video for about five to ten seconds each time. For each video, the tester is asked to hold the gesture for one second before the gesture starts to rotate. Using this principle, 360 videos were shot and used as a template for this experiment.

Once a gesture is detected, the video were marked as a gesture detected, and at the same time recorded which frame contains the gesture for evaluation and detection. The calculation of the detection accuracy rate is described in Eq. (19), where Accuracy is the detection accuracy rate, N is the total number of videos, and CN is the calculation of the average detection time of the number of videos with gestures as shown in Eq. (20), where Time represents the average detection time, CN is the number of videos where gestures are detected, V is the videos where gestures are detected, FrameIdx_v is the video frequency detected in the first frame of video V, where the fps_v rate and average detection seconds are used as evaluation criteria for gesture detection.

$$\text{Accuracy} = \frac{\text{CN}}{N} \quad (19)$$

$$\text{Time} = \frac{1}{\text{CN}} \sum_{v=1}^{\text{CN}} \frac{\text{FrameIdx}_v}{\text{fps}_v} \quad (20)$$

The overall detection and recognition rate is 0.81 s per detection, and the worst recognition rate is 67%, and gestures can be detected in 1.7 s at most shown in Table 7. The overall detection and recognition rate for each objects in the engine room is 89.5%, and the average detection

Table 7
Gesture Detection Experiment Results in Different Positions.

Position	Samsung S8 Recognition Rate (%)	Average Time (s)	Smart Glasses BT-300 Recognition Rate (%)	Average Time (s)
Near Left	100	0.54	80	0.8
Near Middle	97	0.7	73	1.1
Near Right	93	0.82	67	1.7
Far Left	97	0.27	93	0.6
Far Middle	97	0.7	90	1
Far Right	97	0.5	90	0.94
Average	97	0.59	82	1.02

seconds is 0.81 s shown in Table 8. From the experimental results, it can be found that the recognition effect of the engine is the best. The overall detection and recognition rate for each test subjects involved in the experiment is 89.5%, with an average detection rate of 0.82 s shown in Table 9. Experimental results showed that there is a considerable gap between the qualities of the detection results, but none of the testers have poor recognition rates for both lenses. As for the reasons for low recognition rate of tester No. 3 and No. 5, it turns out that whenever testers No. 3 and No. 5 point to the auxiliary water tank, the detection fails (they are all to the left and their gestures are oblique). While the tester themselves are under different cameras, at the same location, and pointing to the same object, Samsung S8 can detect gestures with 100% accuracy. Recognition rate problems were caused by hand tilt.

For videos with detected gestures, the average time it takes to detect a pointing gesture is 0.81 s, which is equivalent to an average of 21 frames of images before the gestures are detected. This means that there are an average of 20 frames of images in front of the video that cannot be detected. In 60% of the videos, gestures were successfully detected in the first frame of the video. It is speculated that the remaining 40% of the videos are due to dynamic motion or pointing gestures and the data set is too different, which makes detection difficult. For the application mode of the system, 0.81 s is still within the acceptable range. Since the recognition rate of gestures is not up to 90%, it can be concluded that the gesture training data set is not comprehensive enough, so 100 gesture images were extracted from the experimental video template to be added to the gesture training data set, and the Mask R-CNN network was retrained.

4.4. Reaction time for gesture detection under different light levels

In order to verify whether the level of light affects the detection of gestures, another experiment was conducted in outdoor-indoor scenarios. Outdoor scenario were shot in an outdoor parking lot, and indoor scenario were shot in an underground parking lot. At the moment of the experiment, the pointing gesture data set of this system covers 400 images, including a total of ten people with both hands, shooting various pointing gesture images at different rotation angles and different distances, the detection accuracy of the training data set is 100%. The experiment involving different light condition invited a total of 13 subjects, using 720 videos as the experimental template. Detection accuracy and average detection seconds are selected as the evaluation criteria for gesture detection. The detection rate is 96.7%, and each detection takes 0.26 s on average. The worst recognition rate is 70%, and gestures can be detected in 1.64 s at most shown in Tables 10 and 11. Gesture captured indoors by smart glasses has poor recognition effect and requires a long detection time. It is presumed that the gesture data set of the system itself is all images taken with a mobile phone under indoor fluorescent lights. In addition, the dark image captured by the smart glasses lens in the room with insufficient light makes it difficult to detect gestures. Although the captured image is affected by light, the detection accuracy is still reaches 87%.

4.5. Gesture and non-gesture detection

The pointing gesture data set of this system covers 400 images,

Table 8
Gesture Detection Experiment Results in Different Objects.

Object	Samsung S8 Recognition Rate (%)	Average Time (s)	Smart Glasses BT-300 Recognition Rate (%)	Average Time (s)
Battery	92	1.17	78	1.63
Engine	100	0.29	95	0.56
Auxiliary Water Tank	98	0.34	73	0.86
Average	97	0.6	82	1.02

Table 9
Gesture Detection Experiment Results in Different Subjects.

Subject	Samsung S8 Recognition Rate (%)	Average Time (s)	Smart Glasses BT-300 Recognition Rate (%)	Average Time (s)
1	94	0.51	89	1.38
2	78	1.1	89	1.18
3	100	0.8	50	1.7
4	100	0.18	89	0.58
5	100	0.49	50	1.48
6	100	0.18	78	0.65
7	94	0.82	100	0.47
8	100	0.33	100	0.36
9	100	0.47	89	2.25
10	100	1.11	89	0.5
Average	97	0.6	82	1.05

Table 10
Outdoor Experiment Results.

Position	Samsung S8 Recognition Rate (%)	Average Time (s)	Smart Glasses BT-300 Recognition Rate (%)	Average Time (s)
Near Left	100	0.03	100	0.08
Near Middle	100	0.06	100	0.11
Near Right	100	0.03	100	0.19
Far Left	100	0.03	100	0.04
Far Middle	100	0.03	100	0.08
Far Right	100	0.03	100	0.18
Average	100	0.04	100	0.11

Table 11
Indoor Experiment Results.

Position	Samsung S8 Recognition Rate (%)	Average Time (s)	Smart Glasses BT-300 Recognition Rate (%)	Average Time (s)
Near Left	100	0.03	93	0.53
Near Middle	100	0.04	87	1.29
Near Right	100	0.1	70	1.64
Far Left	100	0.03	93	0.45
Far Middle	100	0.03	97	0.38
Far Right	100	0.03	80	0.73
Average	100	0.05	87	0.83

including the hands of ten people in various pointing gesture images are shot at different rotation angles and different distances. The detection accuracy of the training data set is 100%. Similar to the experiment design in chapter 3.3, this experiment invites a total of ten test subjects. In order to determine whether the accuracy of gesture detection is affected by the camera lens, this experiment uses the same two cameras as the previous experiments for shooting. The subjects were asked to points to three objects on the device: auxiliary water tank, engine, and battery in six positions in front of the engine room of the car: near left, near middle, near right, far left, middle far, and far right, exactly like the previous two experiments. 360 images are taken as a template for this gesture experiment; the other two camera lenses are used to take 360 images of a simple engine room installation at different positions. The image is used as a template for the experiment without gestures with a total of 720 images. This experiment uses Sensitivity, Specificity, Accuracy, and Kappa indicators to evaluate the reliability of the system. The calculation method is shown in Eqs. (21)–(25), where TP, FP, FN, and TN are defined in Table 12, T is the sum of TP, FP, FN, TN. The Eq.

Table 12
Correlation and Term Definition.

	Positive Correlation	Negative Correlation
Gesture Detected	True Positives (TP) - Positive class defined as positive class - Gesture is defined as gesture	False Positives (FP) - Positive class defined as negative class - No gesture is defined as gesture
Gesture Not Detected	False Negatives (FN) - Negative class defined as positive class - Gesture is defined as no gesture	True Negatives (TN) - Negative class defined as negative class - No gesture is defined as no gesture

(26) is used as the recognition rate of gesture detection, where P represents the recognition rate of gesture detection, N is the total number of gesture images, and CN is the number of images where gestures are detected.

$$\text{Sensitivity} = \frac{TP}{(TP + FN)} \quad (21)$$

$$\text{Specificity} = \frac{TN}{(TN + FP)} \quad (22)$$

$$\text{Accuracy} = \frac{(TP + TN)}{(TP + FP + TN + FN)} \quad (23)$$

$$\text{Confidence} = \frac{(TP + FN) \times (TP + FP)}{T^2} + \frac{(TN + FN) \times (TN + FP)}{T^2} \quad (24)$$

$$\text{KAPPA} = \frac{\text{Accuracy} - \text{Confidence}}{1 - \text{Confidence}} \quad (25)$$

$$P = \frac{CN}{N} \quad (26)$$

The evaluation indicators of the four systems have different meanings. Sensitivity refers to the proportion of gestures correctly detected by the system; specificity refers to the proportion of non-gestures correctly marked as non-gestures by the system; accuracy refers to the correct classification of gestures and non-gestures by the system. Kappa value is used to compare whether the results of two or more observers on the same thing or multiple observations of the same thing by the same observer are consistent (McHugh, 2012). The value of Kappa value is between (-1, 1) where different values have different meanings (Yilmaz & Aktas, 2018). The closer Kappa value is to 1, the system is more reliable. The resulting system evaluation index results are as follows: Sensitivity 93%, specificity 100%, accuracy 96%, and Kappa value 0.93; indicating that the system has a very good reliability shown in Table 13. The recognition rate of images taken with Samsung S8 can reach 97%, the recognition rate of images taken with BT-300 smart glasses can reach 88%, and the recognition rate of the overall gesture detection is as high as 92.5% shown in Tables 14–16. The recognition results of gestures are quite good, and several items even reach 100% recognition rate. Among them, the recognition result of subject No. 6 is poor. After comparison, it turns out that subject no. 6 has lighter skin color, and their hand is more susceptible to light and is harder to detect by the system.

Table 13
Gesture Detection Experiment Results.

Detection Results	Actual Classification	
	Gestures detected	No gestures detected
Gestures detected	334	0
No gestures detected	26	360

Table 14
Gesture Recognition Rate per Objects.

Object	Samsung S8 Recognition Rate (%)	Smart Glasses BT-300 Recognition Rate (%)
Battery	98	88
Engine	98	88
Auxiliary Water Tank	95	88
Average	97	88

Table 15
Gesture Recognition Rate per Position.

Position	Samsung S8 Recognition Rate (%)	Smart Glasses BT-300 Recognition Rate (%)
Near Left	97	87
Near Middle	97	83
Near Right	97	87
Far Left	100	98
Far Middle	100	93
Far Right	93	83
Average	97	88

Table 16
Gesture Recognition Rate per Subject.

Subject	Samsung S8 Recognition Rate (%)	Smart Glasses BT-300 Recognition Rate (%)
1	100	89
2	78	100
3	100	83
4	100	78
5	100	83
6	100	67
7	100	94
8	100	89
9	100	100
10	94	100
Average	97	88

4.6. Finger pointing analysis

This experiment continues with the gesture images detected in the experiments in Sections 3.3 and 3.4 as the experimental template. The image with the detected gestures is analyzed for finger pointing completeness gesture capture. The experimental template contains three light situations, namely indoor fluorescent light and outdoor sunlight. In the context of the gesture experiment template, the mask and skin color detection algorithm detected by Mask R-CNN were compared with the captured results in 3-3 of this paper. The results of gesture capture were divided into five levels, capturing other objects, not grasping the gesture at all, incomplete gesture, slightly broken gesture, and complete gesture.

Every step of finger pointing analysis is very important. If the gesture is not captured properly, it causes errors in pointing analysis. The fingertip point and finger centroid must be extracted through a set of processes. The extraction of fingertip points and finger centroids in the paper requires a more complicated procedure, but the recognition effect can be increased from 58% to 79%. Finally, the research method of this paper is used as a pointing recognition rate experiment. Light affect the use of the system, but the recognition rate of pointing analysis still reaches nearly 80%.

5. Conclusion

This paper proposes a set of augmented reality assistance system which was built on smart glasses as a human-machine interface. It allows users to experience the system from a personal perspective through

the pointing gestures that can show the relevant information of the object pointed by the finger and the corresponding virtual tool. The contributions and novelty by the study can be summarized: (1) improvement for gesture capture due to the error between the analysis angle and the actual angle less than 1.32 degrees; (2) object recognition in general condition reaching as high as 95.5% with an average processing rate at 0.26 s; (3) object recognition in weak light condition still reaching 79%. The findings only use a simple process and a small amount of data to detect specific device objects, by creating a device object data set. Considering that simply using skin color to capture image gestures lose a lot of hand information, the mask information of Mask R-CNN can be used to extract the gesture area in the image based on the skin color information of the current light situation.

The suggestions for the future work are basically due to few current constraints. Although the skin color can be extracted, it is still affected by light. When facing insufficient light, it may affect the gesture detection; thus, pre-processing may need in the future to overcome the light problem. Furthermore, the proposed work is designed for bare hands. It may affect the results for the pointing analysis when wearing accessories on hands such as rings, watches, bracelets, etc., On the other hand, although the pointing gesture data set established by this system covers most types, the pointing gestures based on personal preferences may vary. It is inevitable to face difficulty to identify those various gestures. Analyzing the expansion of the database will complete the work. The other suggestion for future work lies on video and real time implementation. Although all examples and implementation illustrated in Section 4 are based on picture inputs and analysis outputs, the study proves that the proposed system is feasible and efficient in both accuracy rate and processing time. We have conducted a follow-up study using video and real time tests and the results are as good as that in Section 4. We will demonstrate the results with other applications in the future.

CRedit authorship contribution statement

Mu-Chun Su: Conceptualization, Methodology, Supervision. **Jieh-Haur Chen:** Writing - review & editing, Supervision. **Vidya Trisandini Azzizi:** Writing - original draft. **Hsiang-Ling Chang:** Writing - original draft. **Hsi-Hsien Wei:** Writing - review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This paper was partly supported by the Ministry of Science and Technology (MOST), Taiwan, for promoting academic excellent of universities under grant numbers MOST 109-2221-E-008-059-MY3 and MOST 110-2634-F-008-005.

References

Alhaija, H. A., Mustikovela, S. K., Mescheder, L., Geiger, A., & Rother, C. (2018). Augmented reality meets computer vision: Efficient data generation for urban driving scenes. *International Journal of Computer Vision*, 126(9), 961–972.

- Bello, R.-W., Mohamed, A. S. A., & Talib, A. Z. (2021). Contour extraction of individual cattle from an image using enhanced Mask R-CNN instance segmentation method. *IEEE Access*, 9, 56984–57000.
- Carmigniani, J., Furfth, B., Anisetti, M., Ceravolo, P., Damiani, E., & Ivkovic, M. (2011). Augmented reality technologies, systems and applications. *Multimedia Tools and Applications*, 51(1), 341–377.
- Deng, X., Zhang, Y., Yang, S., Tan, P., Chang, L., Yuan, Y.e., & Wang, H. (2017). Joint hand detection and rotation estimation using CNN. *IEEE Transactions on Image Processing*, 27(4), 1888–1900.
- Dongare, Y. B. (2014). Live hand gesture recognition using an android device. *International Journal of Engineering Research and General Science*, 2(6), 363–369.
- Ehkan, P., Siew, S. V., Zakaria, F. F., Warip, M. N. M., & Ilyas, M. Z. (2020). Comparative study of parallelism and pipelining of RGB to HSL colour space conversion architecture on FPGA. *Materials Science and Engineering*, 767(1), Article 012054.
- Gharaibeh, M. K., Gharaibeh, N. K., Khan, M. A., Abu-ain, W. A. K., & Alqudah, M. K. (2021). Intention to use mobile augmented reality in the tourism sector. *Computer Systems Science and Engineering*, 37(2), 187–202.
- Iqbal, A., Basit, A., Ali, I., Babar, J., & Ullah, I. (2021). Automated meter reading detection using inception with single shot multi-box detector. *Intelligent Automation and Soft Computing*, 27(2), 299–309.
- Jang, J., Ko, Y., Shin, W. S., & Han, I. (2021). Augmented reality and virtual reality for learning: An examination using an extended technology acceptance model. *IEEE Access*, 9, 6798–6809.
- Kolkur, S., Kalbande, D., Shimpi, P., Bapat, C., & Jatakia, J. (2017). Human skin detection using RGB, HSV and YCbCr color models. *Proceedings of the International Conference on Communication and Signal Processing*.
- Köpkülü, O., Gunduz, A., Kose, N., & Rigoll, G. (2019). Real-time hand gesture detection and classification using convolutional neural networks. In *Proceedings of 14th IEEE International Conference on Automatic Face and Gesture Recognition*, Lille, France.
- Lee, D., & Lee, S. (2011). Vision-based finger action recognition by angle detection and contour analysis. *ETRI Journal*, 33(3), 415–422.
- Lee, J. Y., & Rhee, G. (2008). Context-aware 3D visualization and collaboration services for ubiquitous cars using augmented reality. *International Journal of Advanced Manufacturing Technology*, 37(5-6), 431–442.
- Lee, K. (2012). Augmented reality in education and training. *TechTrends*, 56(2), 13–21.
- Li, C.-jin., Qu, Z., Wang, S.-ye., & Liu, L. (2021). A method of cross-layer fusion multi-object detection and recognition based on improved faster R-CNN model in complex traffic environment. *Pattern Recognition Letters*, 145, 127–134.
- Loesdau, M., Chabrier, S., & Gabillon, A. (2014). Hue and saturation in the RGB color space. *Proceedings of the International conference on image and signal processing*. Cherbourg, Normandy, France.
- Mchugh, M. L. (2012). Interrater reliability: The kappa statistic. *Biochemia medica*, 22(3), 276–282.
- Nishad, P. (2013). Various colour spaces and colour space conversion. *Journal of Global Research in Computer Science*, 4(1), 44–48.
- Rafique, A. A., Jalal, A., & Kim, K. (2020). Statistical multi-objects segmentation for indoor/outdoor scene detection and classification via depth images. In *Proceedings of 17th International Bhurban Conference on Applied Sciences and Technology*. Islamabad, Pakistan.
- Randive, A., Mali, H., & Lokhande, S. (2012). Hand gesture segmentation. *International Journal of Computer Technology and Electronics Engineering*, 2(3), 125–129.
- Ren, S., He, K., Girshick, R., & Sun, J. (2016). Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6), 1137–1149.
- Ren, Z., Yuan, J., Meng, J., & Zhang, Z. (2013). Robust part-based hand gesture recognition using kinect sensor. *IEEE Transactions on Multimedia*, 15(5), 1110–1120.
- Saravanan, G., Yamuna, G., & Nandhini, S. (2016). Real time implementation of RGB to HSV/HSI/HSL and its reverse color space models. *Proceedings of International Conference on Communication and Signal Processing*.
- Ullo, S. L., Piedimonte, P., Leccese, F., & De Francesco, E. (2019). A step toward the standardization of maintenance and training services in C4I military systems with mixed reality application. *Measurement*, 138, 149–156.
- Xiang, Z., Seeling, P., & Fitzek, F. H. P. (2021). You only look once, but compute twice: Service function chaining for low-latency object detection in softwarized networks. *Applied Sciences*, 11(5), 2177.
- Xie, C., & He, Y. (2016). Spectrum and image texture features analysis for early blight disease detection on eggplant leaves. *Sensors*, 16(5), 676.
- Yilmaz, A. E., & Aktas, S. (2018). Redit and exponential type scores for estimating the kappa statistic. *Kuwait Journal of Science*, 45(1), 89–99.
- Yin, C. Z. Y., Jung, T., Dieck, M. C. T., & Lee, M. Y. (2021). Mobile augmented reality heritage applications: Meeting the needs of heritage tourists. *Sustainability*, 13(5), 2523.