Available online at www.sciencedirect.com

Manufacturing Letters

Manufacturing Letters 35 (2023) 883–894

51st SME North American Manufacturing Research Conference (NAMRC 51, 2023)

Real-Time Human-Computer Interaction Using Eye Gazes

Haodong Chen[a]*, Niloofar Zendehdel[a], Ming C. Leu[a], Zhaozheng Yin[b]

[a]*Department of Mechanical and Aerospace Engineering, Missouri University of Science and Technology, Rolla, MO 65409, USA*
[b]*Department of Biomedical Informatics & Department of Computer Science, Stony Brook University, Stony Brook, NY 117943, USA*

* Corresponding author. Tel.: +1-573-202-0932; fax: +0-000-000-0000. *E-mail address:* h.chen@mst.edu

Abstract

Eye gaze emerges as a unique channel in human-computer interaction (HCI) that recognizes human intention based on gaze behavior and enables contactless access to control and operate software interfaces on computers. In this paper, we propose a real-time HCI system using eye gaze. First, we capture and track eyes using the Dlib 68-point landmark detector, and design an eye gaze recognition model to recognize four types of eye gazes. Then, we construct an instance segmentation model to recognize and segment tools and parts using the Mask Region-Based Convolutional Neural Network (R-CNN) method. After that, we design an HCI software interface by integrating and visualizing the proposed eye gaze recognition and instance segmentation models. The HCI system captures, tracks, and recognizes the eye gaze through a red-green-blue (RGB) webcam, and provides responses based on the detected eye gaze, including the tool and part segmentation, object selection and interface switching. Experimental results show that the proposed eye gaze recognition method achieves an accuracy of > 99% in a recommended distance between the eyes and the webcam, and the instance segmentation model achieves an accuracy of 99%. The experimental results of the HCI system operation demonstrate the feasibility and robustness of the proposed real-time HCI system.

Peer-review under responsibility of the Scientific Committee of the NAMRI/SME.
Keywords: Eye gaze recognition, Human-computer Interaction, Instance Segmentation, Mask R-CNN

1. Introduction

In recent years, human-computer interaction (HCI) has attracted many researchers' interests, as it plays a significant role in the development of artificial intelligence. The HCI is the study of how people interact with computers through interactive systems, such as computer applications and user interfaces. Moreover, various communication channels, such as gestures, speech, electroencephalogram, electromyography, eye gaze, etc., have been used in HCI systems to capture and understand human activities and intentions. Among these channels, the eye gaze is becoming popular for its unique ability to naturally convey the human subject's intention. An HCI system equipped with eye gaze recognition is capable of identifying a human subject's intention based on the subject's eye gaze behavior in human-robot collaboration. As shown in Fig. 1, a human-robot collaboration system using eye gazes mainly consists of two stages: an eye-gaze-based HCI system and a robot operation platform. A human worker can remotely control a robotic arm to choose, deliver, and/or assemble tools and parts from a remote platform using an eye-gaze-based HCI system through a software interface. Webcam *A* in the first stage is set to capture the eye gazes of the human worker, and webcam *B* in the second stage is set to recognize tools and parts on the platform and provide the recognition results to the human worker through the software interface.

In this paper, we mainly consider the first stage of the human-robot collaboration using eye gazes shown on the left side of Fig. 1, i.e., the eye gaze tracking and recognition in an HCI system using a webcam and a software interface. We propose an eye-gaze-based HCI system, which integrates and visualizes three tasks: real-time eye tracking and recognition, object recognition using an instance segmentation model, and tool-and-part selection using the recognized eye gazes.

Peer-review under responsibility of the Scientific Committee of the NAMRI/SME.

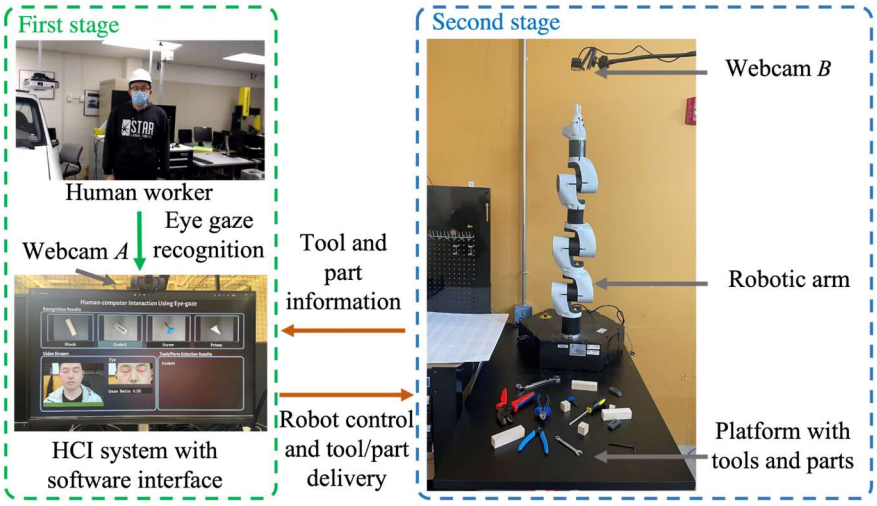


Fig. 1. Human-robot collaboration using eye gazes.

1.1. *Related work*

Human-computer interaction (HCI) is a multidisciplinary field that studies the design, evaluation, and use of computer systems, with a focus on the interactions between people and computers using different communication channels, such as gestures, speech and eye gazes. Qi et. al. proposed an intelligent HCI system using surface EMG gesture signal and used linear discriminant analysis (LDA) and extreme learning machine (ELM) to improve gesture recognition efficiency and accuracy in HCI [1]. Chen et al. designed a software for HCI using dynamic gestures and verbal speech commands, and the software visualized the real-time recognition results of the gestures and speech of human subjects [2]. Eye gaze recognition was initially designed for observing people's gaze behavior during reading tasks in HCI research [3]. Eye gaze recognition devices were usually placed on the top of computers so that they could recognize what users were doing with their eyes while performing different eye gazes. Kyung-Nam et. al. conducted simple eye gaze tracking for HCI through calculating the location of the iris center of an eye [4]. Beymer designed a 3D model with information on the corneal ball, pupil, and fovea areas of the eyes to realize eye tracking and recognition of gaze direction for HCI [5]. Pai et. al. designed simple eye gaze recognition using EMG signals [6]. Later, Gee et. al. developed a flexible vision-based approach, which could estimate the direction of gaze from a single, monocular view of a face [7]. Wang et. al. proposed a system that supported HCIs with head poses, eye gazes, and body gestures, and their method achieved a large performance improvement on the most challenging database at that time [8]. Li et. al. introduced an approach named BayesGaze, which was used to determine the selected target given an eye gaze trajectory [9]. There are also other technologies designed to study eye gaze recognition through machine learning [10], deep learning [11], and virtual reality (VR) [12]. However, the necessity of complicated cameras and/or wearable glasses in tracking eye gazes, and the computational complexity of the eye gaze recognition model limits the real-time application of eye gaze recognition in HCI systems. In this paper, we propose an eye gaze tracking and recognition system through a webcam and effective image processing, which enables robust eye gaze tracking and accurate eye gaze recognition in real time.

Recent theoretical developments have revealed that instance segmentation plays an important part in separating objects into different groups based on their shape and other attributes. Common solutions have been developed to realize instance segmentation tasks, such as Faster-R-CNN [13], Mask R-CNN [14], You Only Look Once (YOLO) [15], etc. However, the lack of visual end-to-end solutions in instance segmentation, such as the applications integrating the pre-trained instance segmentation models and visual interfaces, limits the application of the above approaches in different academic and industrial fields. Solutions without packaged models increase unpredictable difficulty and time cost in the debugging process, which compromises the compatible ability of the existing models to be effectively applied to different tasks. A standard visual end-to-end solution takes the instance segmentation from the beginning to the end. It delivers a complete functional solution without needing to obtain any assistance from a third party or the strong programming background of users. In this paper, we develop a visual end-to-end software interface by integrating the eye gaze tracking and recognition model and the instance segmentation model. The user only uses eye gazes to control the system to execute the instance segmentation task. The software interface simplifies and broadens the application of complex programming models.

1.2. *Contribution of this article*

This paper proposes a real-time HCI system using eye gaze and Mask R-CNN. An overview of the HRC system is depicted in Fig. 2, in which the system recognizes a user's four types of eye gazes in real time, executes instance segmentation of tools and parts in an image, and enables the user to use eye gazes to select the segmented tools and parts through a visual end-to-end software interface.

The contributions of this paper are as follows:

- This paper develops an eye gaze-based HCI system that enables natural and real-time interaction between a human user and a computer using one RGB camera by integrating multiple parallel tasks, including the real-time eye gaze

tracking and recognition, instance segmentation, and visual software interface operation.

- An eye gaze recognition model is designed to use a RGB camera to capture, track and recognize eye movement in real-time.
- An instance segmentation model is trained using the Mask R-CNN to recognize and segment tools and parts in one image.
- A visual end-to-end software interface is designed to visualize the HCI system. The interface is packaged with the models of the eye gaze recognition and tool-and-part recognition and is completely controlled by the proposed eye gazes.

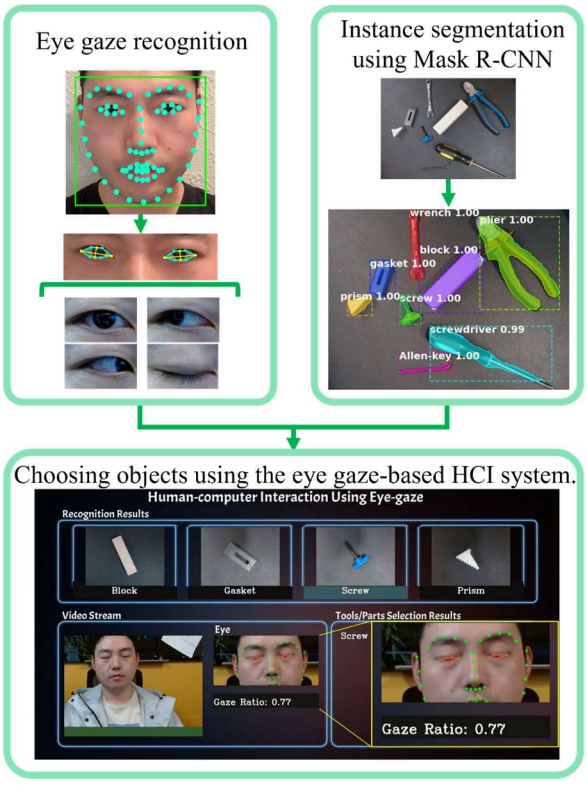


Fig. 2. System overview.

1.3. *Organization of this article*

The rest of this paper is organized as follows: Section 2 describes the design of the eye gaze tracking and recognition model. Section 3 illustrates the instance segmentation of tools and parts. Section 4 designs a visual end-to-end software interface by integrating the proposed eye gaze recognition and instance segmentation models into an intelligent HCI system. Experimental results are shown in Section 5. Section 6 presents the conclusion.

2. Eye gaze recognition

This section introduces our eye gaze tracking and recognition model. Section 2.1 introduces the model of eye detection and tracking, and Section 2.2 details of the real-time eye gaze recognition model.

2.1. *Eye detection and tracking*

Eye detection and tracking are the partial mission of face detection and tracking. The Dlib 68-point facial landmark detector is used for face detection and tracking, which is a pre-trained facial landmark model capable of estimating the locations of 68 coordinates that can be mapped to facial structures (as shown in Fig. 3) [16].

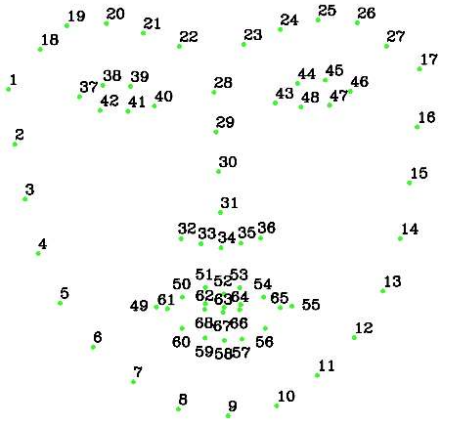
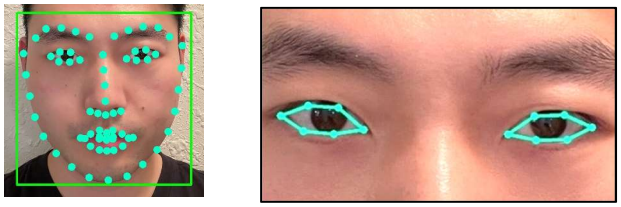


Fig. 3. Dlib 68 facial landmarks.

We first locate a human subject's face in Fig. 4 (a) and map the Dlib 68-point facial landmark on the face. The facial landmark $F_{x,y,w,h}$ is obtained, where the x and y denote the coordinates of the 68 landmarks of the face. The w and h denote the dimensions of the green rectangle shown in Fig. 4 (a), indicating the dimension of the face. After that, we shift the focus from the 68 facial landmarks to the 12 eye landmarks, i.e., landmarks 37-48 in Fig. 3, and extract the coordinates of the 12 landmarks of the left and right eyes in $F_{x,y,w,h}$ as our regions of interest (ROI) and connect them with solid lines, as shown in Fig. 4 (b).



(a) facial landmarks (b) eye landmarks

Fig. 4. Location of face and eyes.

2.2. *Eye gaze recognition model*

As shown in Fig. 5, an eye composes of three main components: pupil, iris, and two side sclera. We focus on the pupil and iris areas' locations, and design four eye gazes shown in Fig. 6 (a-d), including *looking straight ahead*, *looking to the left*, *looking to the right*, and *blinking*. The images in Fig. 6 (a-d) are mirrored. The recognition idea is to use the sclera information to recognize the eye gazes shown in Fig. 6 (a-c) and use the distance between the eyelashes to recognize the eye gaze *blinking* shown in Fig. 6 (d)

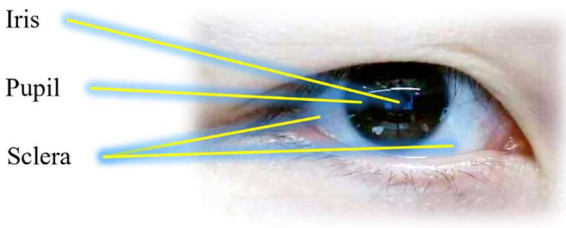
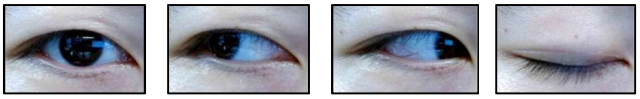


Fig. 5. Eye structure.



(a) *looking straight ahead* (b) *looking to the left* (c) *looking to the right* (d) *blinking*

Fig. 6. Four eye gazes.

To recognize the eye gazes in Fig. 6 (a-c), i.e., *looking straight ahead, looking to the left,* and *looking to the right*, we propose a sclera-ROI-based method shown in Fig. 7 by splitting each eye into two components and calculating the visible sclera areas. Firstly, the eye landmark $E_{x,y}$ is extracted from the facial landmark $F_{x,y,w,h}$. Next, the eye region mapped by $E_{x,y}$ is converted into grayscale and then into binary scale, in which only two color pixels exist, black (pixel value = 0, the iris and pupil regions) and white (pixel value = 1, the sclera region). After that, the number of white pixels, i.e., the sclera region, is counted on both sides of an individual eye as W_{ls} and W_{rs}, where ls and rs represent the left and right sides of an eye, respectively.

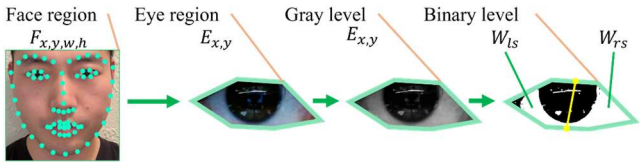


Fig. 7. Feature extraction of the sclera aera of eyes.

An eye's gaze ratio φ is represented as the result of the white pixels of the eye's left part divided by those of the eye's right part, which is shown as follows.

$$\varphi = \frac{W_{ls}}{W_{rs}} \tag{1}$$

A larger eye gaze ratio ρ indicates a more visible sclera on the left part of an eye, and the eye is looking to the right. Otherwise, we assume the eye is looking to the left. Normally both left and right eyes look in the same direction in eye movements. We calculate the average value Φ of the gaze ratios of the left and right eyes as the output to evaluate an eye gaze of a human subject, which is shown as follows:

$$\Phi = \frac{\varphi_L + \varphi_R}{2} \tag{2}$$

where φ_L and φ_R denote the gaze ratios of the left and right eye, respectively. In the 100 experiments of the eye gazes in Fig. 6 (a-c) performed by 2 human subjects (as shown in Fig. 8), we found that the gaze ratio $\Phi < 0.70$ when the subjects' eyes are solidly looking to the left and $\Phi > 1.20$ when the subjects' eyes are solidly looking to the right. If the gaze ratio $\Phi \in [0.70, 1.20]$, the eyes are looking straight ahead.

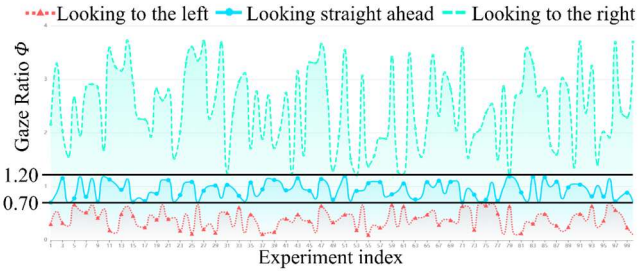


Fig. 8. Gaze ratio distribution of different eye gaze orientations.

To avoid the interference of unconscious eye gaze, such as unconscious left-right glance in the eye gaze recognition, an eye gaze is believed to be valid when the gaze ratio Φ remains < 0.70, $\in [0.70, 1.20]$, or > 1.20 for at least 15 frames, i.e., 0.50 second with a camera frame rate of 30 frames per second (fps).

Regarding the eye gaze *blinking* in Fig. 6 (d), a natural *blinking* gaze occurs when the upper and lower eyelashes are connected and remain for a short period, typically 0.1-0.4 second per blinking for a healthy human subject [17]. To detect the *blinking* gaze, we create a horizontal line l_{ho} and a vertical line l_{ve} for each eye, as shown in Fig. 9, in which the horizontal line l_{ho} connects the left and right landmarks for each eye, i.e., landmarks 37 & 40 and 43 & 46. The vertical line l_{ve} connects the midpoint of the two upper landmarks and the midpoint of the two lower landmarks of each eye, i.e., the midpoints of landmarks 38 & 39 and 41 & 42, and the midpoints of landmarks 44 & 45 and 47 & 48.

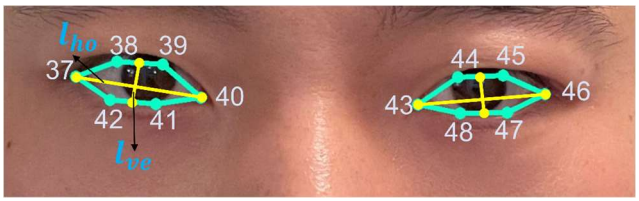


Fig. 9. Eye landmarks and the horizontal and vertical lines.

As shown in Fig. 10 (a) and (b), the length of the horizontal line l_{ho} keeps almost identical when the eyes are opened and closed, while the vertical line l_{ve} is longer when the eyes are open compared with those when they are closed. We take the horizontal line l_{ho} as a reference and calculate the length ratio $l_{ho}/\, l_{ve}$ of an eye as follows:

$$\psi = \frac{l_{ho}}{l_{ve}} \tag{3}$$

where ψ denotes the length ratio of an eye. The result of the $l_{ho}/\, l_{ve}$ provides a larger scale than that of $l_{ve}/\, l_{ho}$ because the l_{ve} is always shorter than l_{ho}. A larger scale yields the results of the length ratio Ψ with a small number of decimal places, i.e., the difference between the results of using $l_{ho}/\, l_{ve}$ is larger than the difference between the results of using $l_{ve}/\, l_{ho}$, which provides convenience and high accuracy in the selection of the length ratio Ψ. We calculate the average value Ψ of the length ratios of the left and right eyes as the output to evaluate a

blinking gaze of a human subject, which is:

$$\Psi = \frac{\psi_L + \psi_R}{2} \quad (4)$$

where ψ_L and ψ_R denote the length ratios of the left and right eye, respectively. Based on 100 eye-closure experiments with two human subjects, we found the average length ratio $\Psi \geq 5.50$ when the subjects closed their eyes. To avoid the interference of normal continuous blinking activity, a *blinking* eye gaze is considered valid when the average length ratio $\Psi \geq 5.50$ for more than 15 frames, i.e., 0.5 second with a camera frame rate of 30 fps, which is longer than the duration time of a typical blinking activity, i.e., 0.10-0.40 second.

(a) opened eyes (b) closed eyes

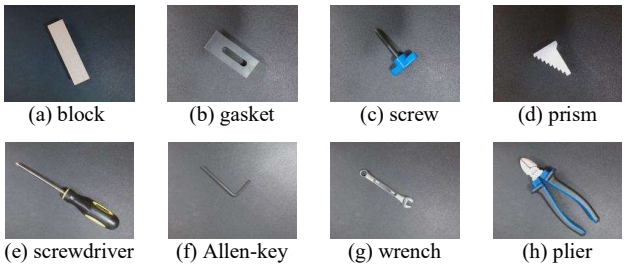
Fig. 10. Length difference between the vertical lines in opened and closed eyes.

3. Instance segmentation using Mask R-CNN

In this section, an instance segmentation model is trained using the Mask R-CNN to recognize and segment tools and parts in the workspace. Section 3.1 describes the dataset construction and data labeling of the tools and parts. Section 3.2 details the model architecture and training process.

3.1. *Data annotation and dataset construction*

We collect 8 tools and parts in our dataset, as shown in Fig. 11. In the image data augmentation, the following techniques are applied, including changing brightness change, flipping, scaling, rotation, and adding Gaussian noise to the images. The dataset includes 370 images, and each image sample includes more than one tool and/or part.



(a) block (b) gasket (c) screw (d) prism

(e) screwdriver (f) Allen-key (g) wrench (h) plier

Fig. 11. Samples of tools and parts.

The quantity of each class is shown in Fig. 12. In the data annotation, we draw polygon masks around the objects of interest, then annotate the classes of the objects, as shown in Fig. 13, in which the polygon masks can extract the object from the image. After that, the classes of objects, image file names, polygon's coordinates, and image dimensions are saved as dataset annotations for the model training. Compared with an object detection model that coarsely localizes multiple objects with bounding boxes, and a semantic segmentation model that produces only pixel-level class labels for each class, an instance segmentation model produces a more meaningful inference for an image, including a segment map of each class as well as each instance of a particular class [18,19].

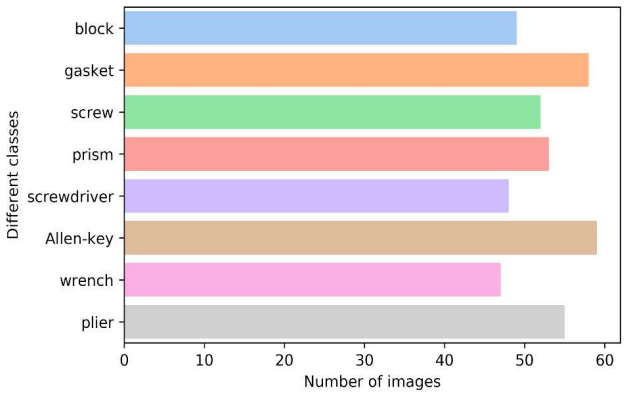
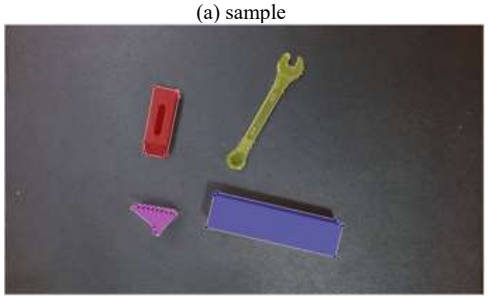


Fig. 12. Distribution of dataset.

(a) sample



(b) sample b

Fig. 13. Samples of data annotation using polygons.

3.2. *Instance segmentation model architecture*

The architecture of our instance segmentation model using the Mask R-CNN is shown in Fig. 14. The size of the input image is 640×480, and the model shifts the focus to the ROI of all object pixels in the image. In the training progress, the Mask R-CNN generates masks of objects in an input image using the weights of the pre-trained ResNet-50 network on the COCO dataset [20], which allows us to perform robust instance segmentation and classification without having to retrain our custom weights in generating masks. The input image is fed into the deep neural network, which consists of several convolutional and fully connected layers. The convolutional layers extract low-level features, such as edges and textures of the tools and parts in the image, while the fully connected layers extract high-level features representing the tools and

parts in the image. These features are then fed into the region proposal network (RPN), which generates a set of region proposals for each object in the image. The region proposals are then fed into the ROI pooling layer, which extracts features from each region proposal and passes them to the classification and mask heads. The classification head outputs the class probabilities for each region proposal, while the mask head generates the masks for each object in the image. The features extracted by the deep neural network are used to differentiate the objects in the image and generate the masks [21-23].

There are mainly two parallel output branches, in which the first branch in Fig. 14 returns the class labels and bounding box coordinates for each object in the input image. In the second branch, the model predicts the segmentation masks of each object, i.e., the different polygons with different colors shown in Fig. 14, and draws the masks on each ROI to provide visual segmentation results. Particularly, the second branch applies a small fully convolutional network (FCN) to each ROI and predicts a segmentation mask in a pixel-to-pixel manner, enabling a rapid training process. Finally, the instance segmentation model produces a segment map of each class and each tool/part of a particular class as inferences. The experimental results of the instance segmentation model are shown in Section 5.2.

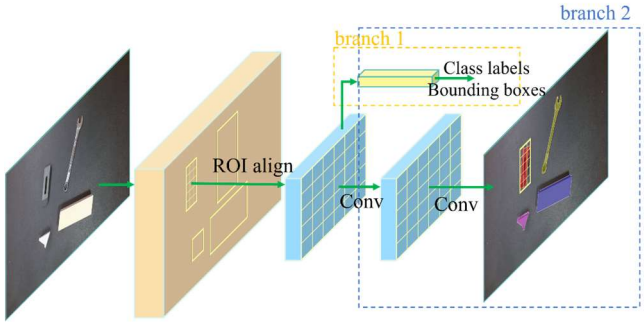


Fig. 14. Architecture of the Mask R-CNN model (Conv: convolutional layer).

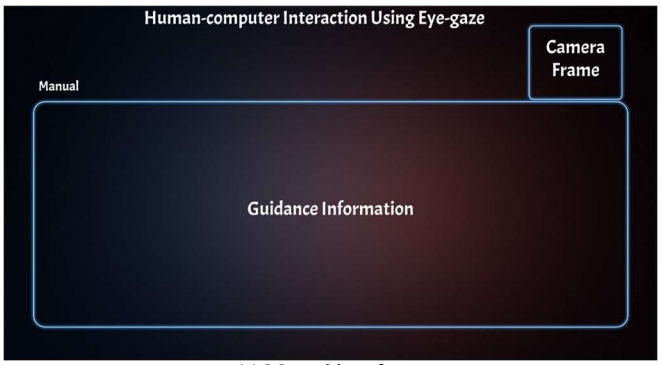
4. System integration and visualization

A software interface is developed to enable a visual end-to-end solution for our HCI system. A human user can perform instance segmentation and choose multiple tools and parts using only eye gazes [24, 25]. The system operates with the following mechanism:

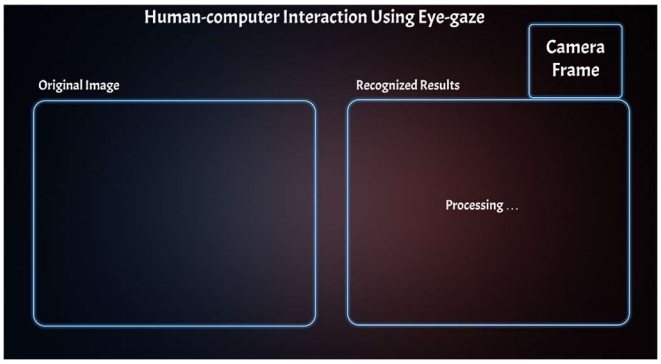
1) When the system starts, the system's user manual is displayed to provide instructional information to the user (Fig. 15 (a)). After that, the system turns on the webcam to detect the user's *blinking* eye gaze. The user can blink his/her eyes to continue.

2) Next, the instance segmentation interface is displayed as shown in Fig. 15 (b), in which the left window of the interface displays a random image with multiple tools and parts, and the right window of the interface displays the instance segmentation results of the input image. Then the webcam is turned on to detect the user's *blinking* eye gaze, and the software interface moves to the next step if the user blinks his/her eyes.

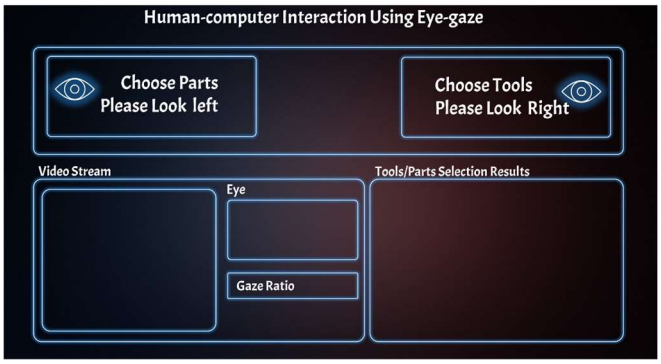
3) After that, the interface shown in Fig. 15 (c) is displayed, in which the user can choose tools or parts by looking to the left or right, respectively. Then, the segmented tools or parts are listed on the top windows shown in Fig. 15 (d), along with the recognized class labels, which are cyclically highlighted individually. The user can select a highlighted tool or part by blinking. Each highlighted label remains for 1.00 second. After choosing the tool or part, the system rolls back to the interface shown in Fig. 15 (c), allowing the user to continue choosing other tools and parts. The label of the selected tool or part is saved on the board on the right window of the interface shown in Fig. 15 (c) and (d).



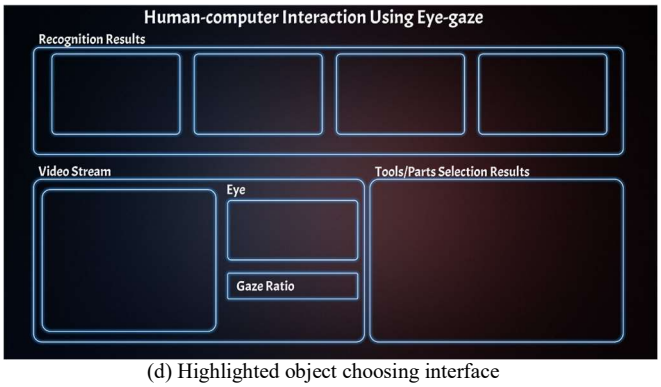
(a) Manual interface



(b) Instance segmentation interface



(c) Tools and parts choosing interface



(d) Highlighted object choosing interface

Fig. 15. Software interface for the HCI using eye gazes.

5. Experiments

In this section, we evaluate the proposed eye gaze tracking and recognition model in Section 2, the instance segmentation model in Section 3, and the software interface in Section 4.

5.1. *Experimental results of eye gaze recognition*

To evaluate the accuracy of the proposed eye gaze recognition model, we conduct experiments involving 2 subjects. The distance between the eyes and the webcam is around 40 cm, and the webcam frame rate is 30 fps. The performance of the eye gaze recognition model is shown in Table 1, in which the ground truth of the eye gazes, and the recognition results are given, such as the 92/95 in the eye gaze *blinking* indicates that 92 *blinking* eye gazes are recognized out of the 95 *blinking* eye gazes in the ground truth. The average accuracy of the experimental results for the two subjects is 98.90%, and the computation time of an eye gaze is less than 0.001 second on average, which is faster than real-time. The quantity results of the eye gazes *looking to the left* and *looking to the right* of the two subjects are shown in the Fig. 16.

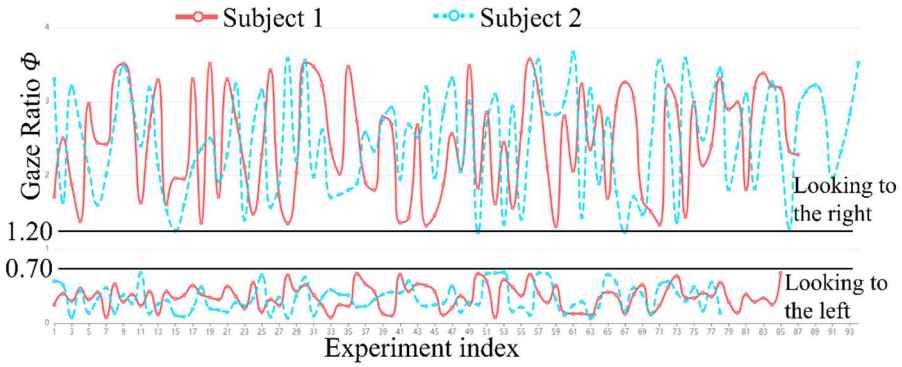


Fig. 16. Experimental results of looking to the left and looking to the right.

Table 1 Performance of the eye gaze recognition model.

Subject	Eye gazes				Average accuracy
	looking straight ahead	*looking to the left*	*looking to the right*	*blinking*	
1	97/97	85/85	86/87	92/95	98.92%
2	103/103	77/78	92/94	90/91	98.87%

We conduct relevant experiments to evaluate the effect of distance variation between the eyes and the webcam on eye gaze tracking and recognition. The experimental results are shown in Fig. 17, in which the distance between the human subject and the webcam increases from about 10 cm to about 160 cm, and the human subject's eyes are successfully detected and tracked with the eye landmarks shown in Fig. 9. Note that the gaze landmarks keep the same size when the distance between the human and the camera continuously increases, resulting in the overlapping of the eye and the landmarks in Fig. 17 (d) and (e), where the sizes of the eyes are smaller than the eyes in Fig. 17 (a-c) because of the long distance between the eyes and the camera. The processing time of eye tracking is less than 0.001 second, which is real-time. The experimental results indicate the robustness of the real-time eye tracking.

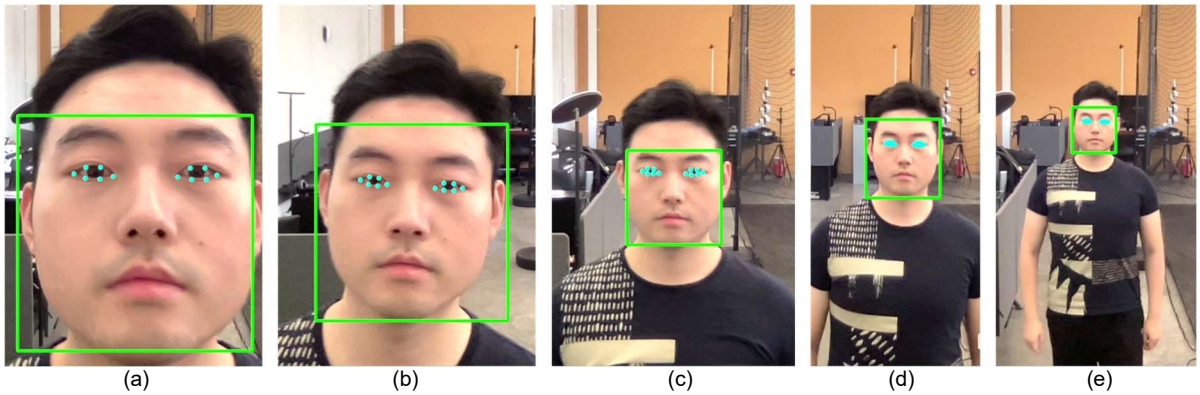


Fig. 17. Eye tracking at different distances between the eyes and the webcam

The experimental results shown in Table 2 indicate the performance of the eye gaze recognition model when dealing with different distances between the human eyes and the webcam. Each distance case includes 100 experiments for each

eye gaze in the first column in Table 2. The accuracy values of all eye gazes decrease as the distance between the subject's eyes and the webcam increases, but at a distance of about 160 cm between the subject's eyes and the webcam, the accuracy of the eye gaze *looking straight ahead* remains 99%, the accuracy of the eye gazes *looking to the left* and *looking to the right* remains 97%. The accuracy of the eye gaze blinking remains 94%. The experimental results illustrate the robustness of the proposed eye gaze recognition model in handling different distances between the human subject's eyes and the webcam.

Table 2 Performance (%) of the eye gaze recognition model in dealing with different distances between eyes and webcam.

Eye gaze	10.00-40.00cm	40.00-70.00cm	70.00-100.00cm	100.00-130.00cm	130.00-160.00cm
looking straight ahead	100.00	100.00	100.00	99.00	99.00
looking to the left	100.00	100.00	97.00	97.00	97.00
looking to the right	100.00	99.00	98.00	97.00	97.00
blinking	100.00	99.00	98.00	96.00	94.00

It has been implied that the distance between a human subject's eyes and a webcam in eye gaze recognition is approximately 40-60 cm [26]. According to the Occupational Safety and Health Administration, the recommended safe viewing distance between the eyes and a computer monitor is around 40-70 cm [27]. The webcam is fitted with a computer monitor in our eye gaze recognition. Therefore, the user's eyes are at the same distance from the webcam and the monitor. When the distance between eyes and the webcam is 40-60 cm, i.e., the second column in Table 2, our recognition accuracy achieves 100% for the eye gazes *looking straight ahead* and *looking to the left*, and 99% for the eye gaze *looking to the right* and *blinking*, which enables accurate recognition within the safe distance between the eyes to the monitor/ webcam.

Overall, the proposed eye gaze recognition model can maintain robust eye tracking within a distance of 160 cm between a subject's eyes and a webcam and achieve an average eye gaze recognition accuracy of 99.5% within the recommended safe distance (40-60 cm) between a subject's eyes and a webcam. The proposed method can recognize an eye gaze within less than 0.001 second under 30 frame-per-second, which is much faster than real-time (1/30 second, i.e., around 0.333 second). The proposed gaze recognition model only needs a simple RGB camera to collect gaze data, which enables our model to be used in other application platforms by only adding one RGB camera and shows the generalization ability of our method.

5.2. *Experimental results of instance segmentation*

The dataset includes 370 images of 8 different tools and parts, and each image includes more than one object. The ratio of the training, the validation, and the testing dataset is 6:2:2. In the training process, the epoch is 100, and the learning rate is 0.001. Several widely used metrics are used to evaluate the classification performance:

$$Accuracy = \frac{TP+TN}{TP+FN+FP+T} \tag{5}$$

$$Precision = \frac{TP}{TP+FP} \tag{6}$$

$$Recall = \frac{TP}{TP+F} \tag{7}$$

$$F_1score = 2 \cdot \frac{Precision \cdot Recall}{Precision+Re} \tag{8}$$

where True Positive (*TP*) refers to a sample x belonging to a class *C* that is correctly classified as *C*. True Negative (*TN*) indicates that a sample x from a 'not *C*' class is correctly classified as a member of the 'not *C*' class. The False Positive (*FP*) is when a sample x from a 'not *C*' class is incorrectly classified as class *C*. The False Negative (*FN*) denotes that a sample x from class *C* is misclassified as belonging to a 'not *C*' class. The F_1score is the harmonic mean of precision and recall and provides a single value that summarizes the accuracy of the model in recognizing objects.

The confusion matrix of the test dataset is shown in Fig. 18, in which the rows represent the ground truth labels, and the columns represent the recognition results. The values along the confusion matrix diagonal show the recognition accuracy of each class. All accuracy values are larger than 98%, and the accuracy of the label *plier*, *block*, and *prism* is 100%.

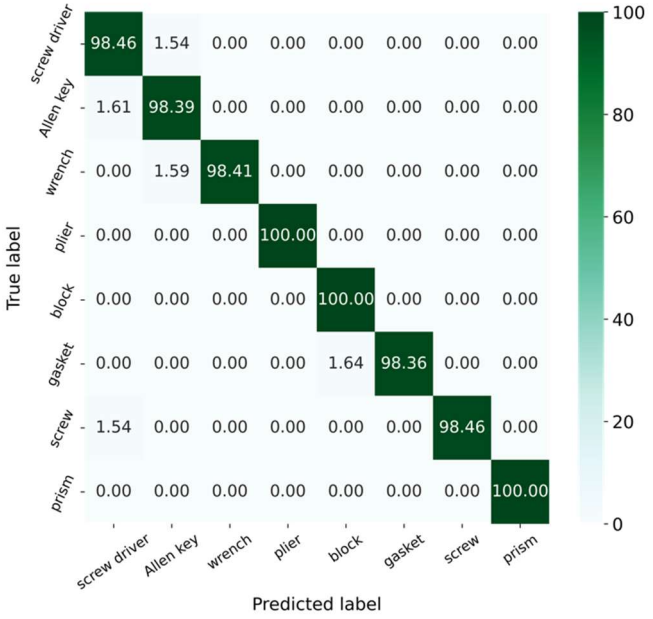


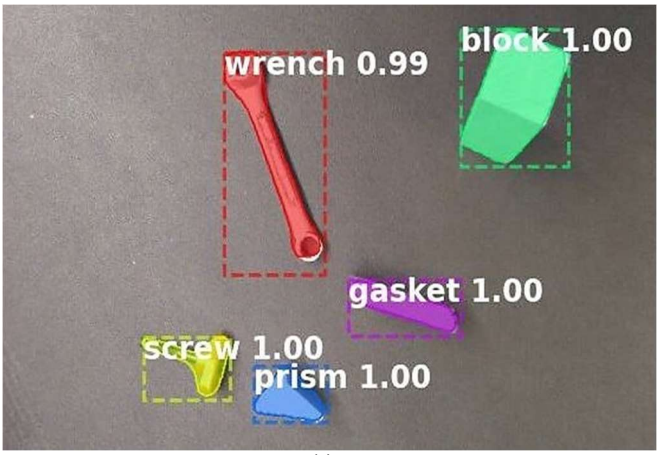
Fig. 18. Confusion matrix of the instance segmentation model (%).

Table 3 shows the value of the metrics shown in Eqs. (5-8), i.e., *Accuracy, Precision, Recall, and* F_1score. The values of these four metrics are all larger than 96% for the 8 labels. The average values of the *Accuracy, Precision, Recall, and* F_1score are larger than 99% for all of them. The metrics of the *plier* and the *prism* are all 100%. The recognition results show the performance of the trained instance segmentation model in segmentation of the 8 tools and parts, and indicate high accuracy of the instance segmentation model.

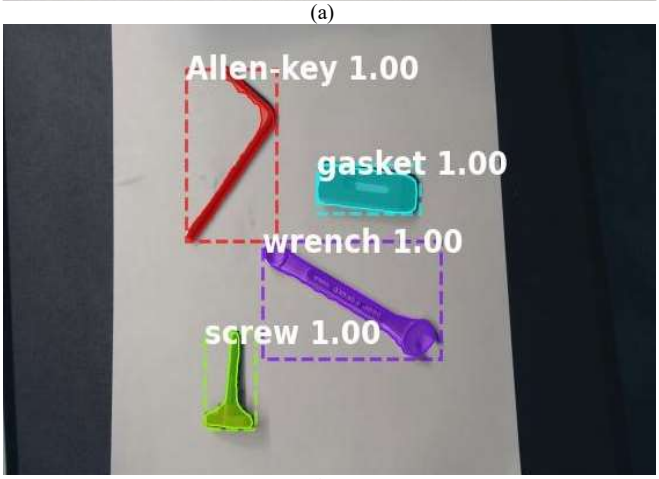
Table 3 Performance (%) of instance segmentation model.

Labels	Accuracy	Precision	Recall	F_1score
Screwdriver	98.46	96.97	98.46	97.71
Allen-key	98.39	96.83	98.39	97.60
Wrench	98.41	100.00	98.41	99.20
Plier	100.00	100.00	100.00	100.00
Block	100.00	98.36	100.00	99.17
Gasket	98.36	100.00	98.36	99.17
Screw	98.46	100.00	98.46	99.22
Prism	100.00	100.00	100.00	100.00

Two samples of the instance segmentation are shown in Fig. 19 (a, b), in which the bounding boxes, masks, class labels, and recognition confidence scores for all objects are shown respectively. The bounding boxes and the masks of an object in the image share the same color, and the bounding boxes are given in dashed line. The confidence score in the interval [0, 1] represents the confidence that an object is recognized as the given class [28]. All recognition confidence scores are higher than 99.30%, and all predicted masks and bounding boxes correctly match the target objects.



(a)



(b)

Fig. 19. Samples of instance segmentation results.

When processing an input image with multiple identical objects, the instance segmentation model generates segmentation maps of each class and each instance of a particular class, as shown in Fig. 20. Although the polygon masks and the object shapes in Fig. 20 are not 100% identical, the two identical blocks and two identical gaskets are individually recognized with correct bounding boxes, class labels, and recognition confidence scores of ≥ 92%.

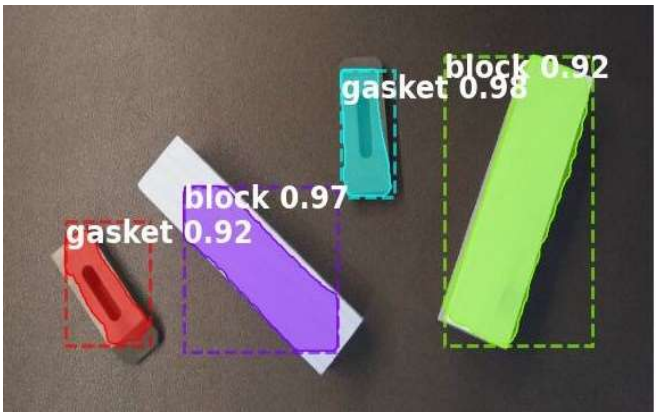


Fig. 20. A sample of instance segmentation with multiple identical objects.

5.3. *Experimental results of system visualization*

When the system starts, the manual interface is displayed, as shown in Fig. 21, which provides the system's user manuals for the eye-gaze-based software, including guidance and precautions during the usage of the software, such as choosing tools by looking to the left, choosing parts by looking to the right, keeping the head still and being relaxed, and only move eyes to look to the left or right sides. etc. The manuals are given line-by-line. After all the manuals are given, the webcam is turned on to capture the *blinking* eye gaze.

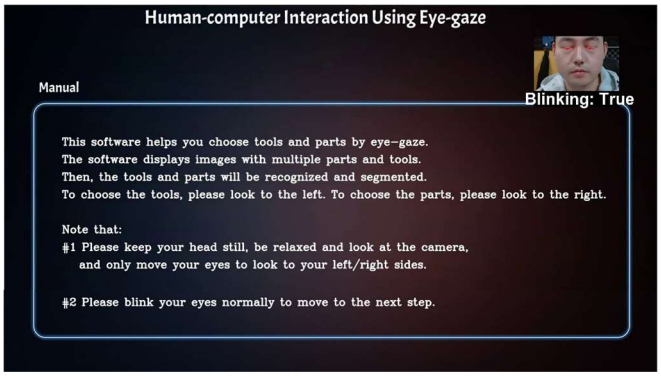


Fig. 21. Visual manual information.

The visual instance segmentation interface is shown in Fig. 22, in which the details of the bounding boxes, masks, class labels, and confidence scores are given. The tools and parts are correctly segmented with bounding boxes and recognized with confidence scores ≥ 99%. The instance segmentation interface provides the user with visual information about the tools and parts on the platform, and it is capable of recognizing the *blinking* eye gaze of the user, as shown in the right corner of Fig. 22.

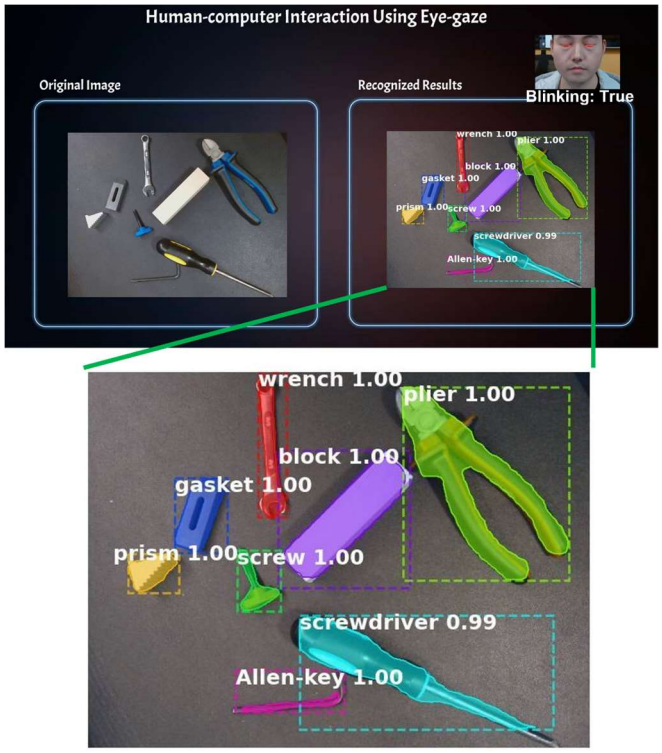


Fig. 22. Visual instance segmentation.

The interface shown in Fig. 23 is displayed when the *blinking* eye gaze is detected in the visual instance segmentation shown in Fig. 22. The RGB webcam frames and the user's facial and eye landmarks are displayed in Fig. 23 simultaneously. Based on the options on the top of the interface, the user in Fig. 23 is looking to the left with a gaze ratio of 0.25, which indicates the user's intention to choose the segmented parts in Fig. 22.

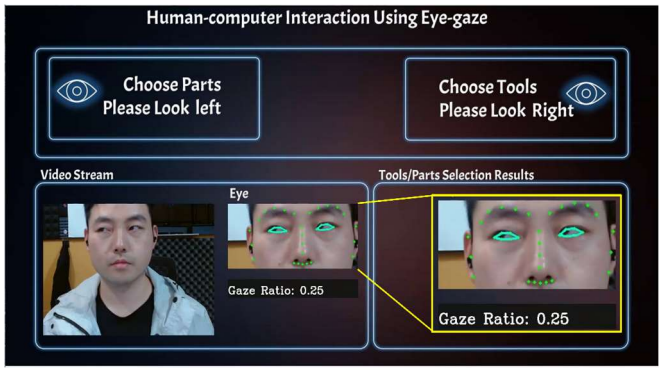


Fig. 23. Visual eye gaze recognition - *looking to the left*.

After that, the interface shown in Fig. 24 cyclically highlights each part for 1 second and saves the chosen part *gasket* on the board when the software interface recognizes the user's *blinking* eye gaze. A dynamic green bar is displayed at the bottom of the webcam frame window to indicate the *blinking* recognition. This bar dynamically grows longer from left to right as the average length ratio Ψ of the *blinking* increases. A full-length green bar and red highlighted eye landmarks in Fig. 24 indicate a valid *blinking* eye gaze.

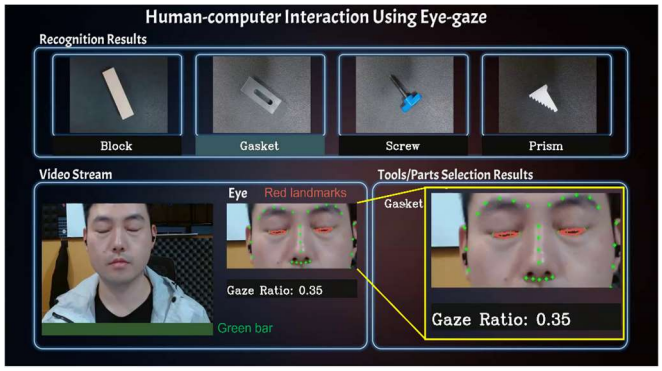
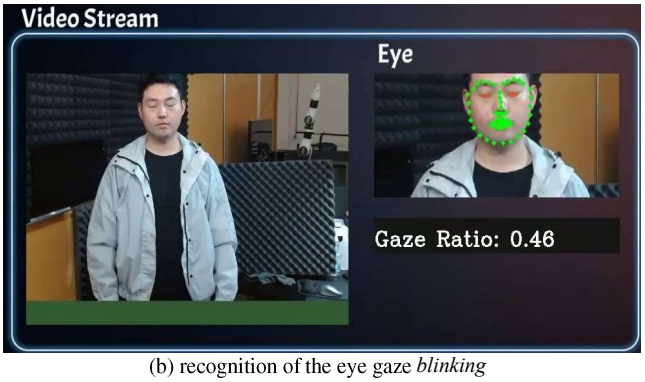


Fig. 24. Visual eye gaze recognition - choosing a part.

The eye gaze recognition at a distance of about 160 cm between the user and the webcam is shown in Fig. 25 (a, b), in which the eye gaze *looking to the left* is recognized with a gaze ratio of 0.46 (Fig. 25 (a)). The eye gaze *blinking* is recognized with the full-length green bar at the bottom of the left RGB webcam frame and the red highlighted eye landmarks system (Fig. 25 (b)). The results in Fig. 25 (a, b) show the robustness of the system in handling long distances between the user and the webcam.

The experimental results show that the visual end-to-end software interface can automatically perform tasks, including the eye gaze tracking and recognition, instance segmentation of tools and parts, and dynamic interface switching with the integrated eye gaze recognition and instance segmentation models. A video demonstration is available: link.

(a) recognition of the eye gaze *looking to the left*



(b) recognition of the eye gaze *blinking*

Fig. 25. Visual eye gaze recognition in around 160 cm distance between the eyes and the webcam.

6. Conclusion

In this paper, we propose an eye-gaze-based human-computer interaction (HCI) system enabling real-time interaction between a human subject and a visual software interface using a webcam. In the proposed HCI system, an eye gaze recognition model is designed using the distribution of the sclera region of the eyes and the distance between the upper and lower eyelashes of the eyes. An instance segmentation model is proposed to recognize and segment tools and parts using the Mask Region-Based Convolutional Neural Network (R-CNN) approach. A visual software interface is designed by integrating the eye gaze recognition and instance segmentation models.

The proposed HCI system enables the real-time capture, tracking, and recognition of four eye gazes via a webcam. It can execute the recognition of tools and parts as well as provide a customized selection of objects based on the recognized eye gazes. According to the experimental results of our models, the proposed eye gaze recognition model achieves an average accuracy of 99% within a recommended safe distance (40-60 cm) between the eyes and the webcam, and the instance segmentation model achieves an average accuracy of 99%. The real-time system experimental results demonstrate the feasibility and robustness of the proposed HCI system.

In the future, we will consider the following studies: 1) More eye gaze data under different environmental factors, such as lightning conditions, reflections and shadows, will be considered to increase the robustness of the gaze recognition model. 2) Various backgrounds will be considered in the instance segmentation of tools and parts to yields the generalizability of the instance segmentation model. 3) A human-robot collaboration system will be constructed by applying the proposed HCI system to the collaboration between a human subject and a robot.

Acknowledgements

This research work was financially supported by the National Science Foundation grants CMMI-1646162 and CMMI-1954548 and also by the Intelligent Systems Center at Missouri University of Science and Technology. Any opinions, findings, conclusions, and recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] Qi, J., Jiang, G., Li, G., Sun, Y., & Tao, B. (2019b). Intelligent Human-Computer Interaction Based on Surface EMG Gesture Recognition. IEEE Access, 7, 61378–61387. https://doi.org/10.1109/access.2019.2914728

[2] Chen, H., Leu, M. C., & Yin, Z. (2022). Real-Time Multi-Modal Human–Robot Collaboration Using Gestures and Speech. Journal of Manufacturing Science and Engineering, 144(10). https://doi.org/10.1115/1.4054297

[3] Mahmud, S., Lin, X., & Kim, J. H. (2020). Interface for Human Machine Interaction for assistant devices: A Review. 2020 10th Annual Computing and Communication Workshop and Conference (CCWC). https://doi.org/10.1109/ccwc47524.2020.9031244

[4] Kim, K.N. and Ramakrishna, R.S. (1999). Vision-based eye-gaze tracking for human computer interface. IEEE SMC'99 Conference Proceedings. 1999 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No.99CH37028). https://doi.org/10.1109/icsmc.1999.825279

[5] Beymer, D., & Flickner, M. D. (2003). Eye gaze tracking using an active stereo head. Computer Vision and Pattern Recognition. https://doi.org/10.1109/cvpr.2003.1211502

[6] Pai, Y. S., Dingler, T., & Kunze, K. (2019). Assessing hands-free interactions for VR using eye gaze and electromyography. Virtual Reality, 23(2), 119–131. https://doi.org/10.1007/s10055-018-0371-2

[7] Hutchinson, T., White, K., Martin, W., Reichert, K., & Frey, L. (1989). Human-computer interaction using eye-gaze input. IEEE Transactions on Systems, Man, and Cybernetics, 19(6), 1527–1534. https://doi.org/10.1109/21.44068

[8] Wang, K. L., Zhao, R., & Ji, Q. (2018). Human Computer Interaction with Head Pose, Eye Gaze and Body Gestures. IEEE International Conference on Automatic Face & Gesture Recognition. https://doi.org/10.1109/fg.2018.00126

[9] Li, Z., Zhao, M., Wang, Y., Rashidian, S., Baig, F., Liu, R., Liu, W., Beaudouin-Lafon, M., Ellison, B., Wang, F., & Bi, X. J. (2021). BayesGaze: A Bayesian Approach to Eye-Gaze Based Target Selection. Proceedings, 2021, 231–240. https://doi.org/10.20380/gi2021.35

[10] Kornhuber, M., Dunst, S. (2022). Automated Classification of Cellular Phenotypes Using Machine Learning in Cellprofiler and CellProfiler Analyst. In: Zi, Z., Liu, X. (eds) TGF-Beta Signaling. Methods in Molecular Biology, vol 2488. Humana, New York, NY. https://doi.org/10.1007/978-1-0716-2277-3_14

[11] Klaib, A. F., Alsrehin, N. O., Melhem, W. Y., Bashtawi, H. O., & Magableh, A. A. (2021). Eye tracking algorithms, techniques, tools, and applications with an emphasis on machine learning and Internet of Things technologies. Expert Systems With Applications, 166, 114037. https://doi.org/10.1016/j.eswa.2020.114037

[12] Rahman, Y., Asish, S. M., Fisher, N. S., Bruce, E. C., Kulshreshth, A. K., & Borst, C. W. (2020). Exploring Eye Gaze Visualization Techniques for Identifying Distracted Students in Educational VR. 2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR). https://doi.org/10.1109/vr46266.2020.00009

[13] Girshick, R., (2015). Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision (ICCV) (pp. 1440-1448). https://doi.org/10.1109/iccv.2015.169

[14] He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 2961-2969). https://doi.org/10.48550/arXiv.1703.06870.

[15] Wang, C.Y., Bochkovskiy, A. and Liao, H.Y.M., (2022). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. arXiv preprint arXiv:2207.02696. https://doi.org/10.48550/arXiv.2207.02696

[16] King, D. (2009). Dlib-ml: A Machine Learning Toolkit. Journal of Machine Learning Research, 10(60), 1755–1758.

[17] Schiffman, H. R. (2000). Sensation and Perception: An Integrated Approach. Wiley. 45 - 47

[18] Guo, Y., Liu, Y., Georgiou, T., & Lew, M. S. (2018). A review of semantic segmentation using deep neural networks. International Journal of Multimedia Information Retrieval, 7(2), 87–93. https://doi.org/10.1007/s13735-017-0141-z

[19] Chen, H., Teng, Z., Guo, Z., & Zhao, P. (2020). An Integrated Target Acquisition Approach and Graphical User Interface Tool for Parallel Manipulator Assembly. Journal of Computing and Information Science in Engineering, 20(2). https://doi.org/10.1115/1.4045411

[20] Hafiz, A. M., & Bhat, G. M. (2020). A survey on instance segmentation: state of the art. International Journal of Multimedia Information Retrieval, 9(3), 171–189. https://doi.org/10.1007/s13735-020-00195-x

[21] Tao, W., A., Chen, H., Leu, M., Yin, Z., & Qin, R. (2019). Real-Time Assembly Operation Recognition with Fog Computing and Transfer Learning for Human-Centered Intelligent Manufacturing. Procedia Manufacturing, 48, 926–931. https://doi.org/10.1016/j.promfg.2020.05.131

[22] Tao, W., Chen, H., Moniruzzaman, M., Leu, M., Yi, Z., & Qin, R. (2021). Attention-Based Sensor Fusion for Human Activity Recognition Using IMU Signals. Cornell University - ArXiv. https://doi.org/10.48550/arxiv.2112.11224

[23] Teng, Z., Chen, H., Hou, Q., Song, W., Gu, C., & Zhao, P. (2020, November). Design of a Cognitive Rehabilitation System Based on Gesture Recognition. In ASME International Mechanical Engineering Congress and Exposition (Vol. 84522, p. V005T05A067). American Society of Mechanical Engineers. https://doi.org/10.1115/IMECE2020-23579

[24] Chen, H., Zhu, H., Teng, Z., & Zhao, P. (2020). Design of a Robotic Rehabilitation System for Mild Cognitive Impairment Based on Computer Vision. Journal of Engineering and Science in Medical Diagnostics and Therapy. https://doi.org/10.1115/1.4046396

[25] Chen, H., Wang, Y., Caboni, M. F., Chen, W., & Zhao, P. (2018). A GUI Software for Automatic Assembly Based on Machine Vision. International Conference on Mechatronics. https://doi.org/10.1109/icmra.2018.8490562

[26] Modi, N., & Singh, J. (2020). A Review of Various State of Art Eye Gaze Estimation Techniques. Advances in Intelligent Systems and

Computing, 501–510. https://doi.org/10.1007/978-981-15-1275-9_41
[27] Bali, J., Neeraj, N., & Bali, R. T. (2013). Computer vision syndrome: A review. Journal of Clinical Ophthalmology and Research, 2(1), 61. https://doi.org/10.4103/2320-3897.122661
[28] Chen, H., Leu, M., Tao, W., & Yin, Z. (2020). Design of a Real-Time Human-Robot Collaboration System Using Dynamic Gestures. ASME 2020 International Mechanical Engineering Congress and Exposition. https://doi.org/10.1115/imece2020-23650