A Novel Gaze-Point-Driven HRI Framework for Single-Person

Wei Li[1], Pengfei Yi[1], Dongsheng Zhou[1,2(✉)], Qiang Zhang[1,2], Xiaopeng Wei[2], Rui Liu[1], and Jing Dong[1]

[1] Dalian University, Dalian, People's Republic of China
zhouds@dlu.edu.cn
[2] Dalian University of Technology, Dalian, People's Republic of China

Abstract. Human-robot interaction (HRI) is a required method of information interaction in the age of intelligence. The new human-robot collaboration work mode is based on this information interaction method. Most of the existing HRI strategies have some limitations: Firstly, limb-based HRI relies heavily on the user's physical movements, making interaction impossible when physical activity is limited. Secondly, voice-based HRI is vulnerable to noise in the interaction environment. Lastly, while gaze-based HRI reduces the reliance on physical movements and the impact of noise in the interaction environment, external wearables result in a less convenient and natural interaction process and increase costs. This paper proposed a novel gaze-point-driven interaction framework using only RGB cameras to provide a more convenient and less restricted way of interaction. At first, gaze points are estimated from images captured by cameras. Then, targets can be determined by matching these points and positions of objects. At last, objects gazed at by an interactor can be grabbed by the robot. Experiments under conditions of different lighting, distances, and different users on the Baxter robot show the robustness of this framework.

Keywords: Human-robot interaction · Gaze point · Grab

1 Introduction

HRI is ubiquitous, and especially the new human-robot collaborative work model is inseparable from the interaction between workers and robots. There is a wide range of interaction methods such as touch (touch display control), voice (voice input control), gesture-based, gaze-based, etc. Touch [1], voice [15], and gesture-based interaction [13,22] are currently the most common methods of HRI. However, these three interaction methods have some limitations, such as being susceptible to the interaction environment (e.g., interaction distances, noise in the

H. Gao and X. Wang (Eds.): CollaborateCom 2021, LNICST 406, pp. 661–677, 2021.
https://doi.org/10.1007/978-3-030-92635-9_38

environment), relying heavily on the user's body movements, and contact interaction increases the risk of spreading some diseases. In the actual human-robot collaboration process, there may be situations where workers and robots cannot interact through language due to the noise in the work environment, and busy workers cannot operate the robot through their limbs. To improve the efficiency of human-robot collaboration and make human-robot interaction more natural, smooth, and safer, so nowadays, gaze-point-based human-robot interaction is increasingly used.

Most current gaze-point-based HRI methods require the help of external wearables (e.g., eye-tracking devices). Although external wearables, such as eye-tracking devices, are inexpensive, they have achieved good experimental results. However, there are still many disadvantages such as calibration problems, inconvenience in carrying, and influence by other glasses (e.g., myopic glasses). Collaborators wear eye-tracking for collaboration, which not only increases costs but also inconveniences collaboration.

To overcome the drawback of external wearables in HRI, improve the convenience of HRI based on gaze points, and make HRI more intelligent and humanized. Inspired by human gaze behavior, we build a framework for gaze-point-driven interaction without the assistance of any external wearables. In this framework, the robot uses only RGB cameras to infer the gaze target from the user's gaze points and then actively grasp it. Our main work in this paper is as follows:

- We build an HRI framework in which the robot infers the gaze target by gaze points and actively grasps it without the aid of any external wearables, significantly improving the ease and reducing the cost of interaction.
- We design a filter to filter invalid gaze points, thus enabling the robot to reason more accurately about the user's gaze target.
- We propose an efficient sorting-based entity matching method that does not rely on object features and does not require the processing of object features.

2 Related Work

In the gaze-point-based HRI approach, gaze point estimation is the first step in the interaction. Then uses the estimated gaze points to drive the robot through the various tasks in HRI. This section provides an overview of the methods commonly used for gaze point estimation and their application in HRI.

2.1 Gaze Point Estimation

In the current research, the methods of acquitting gaze points can divide into those that rely on external wearables and those that do not. Among the methods with external wearables, the main one is the use of eye-tracking. While among the methods without external wearables, deep learning-based methods are the dominant ones.

Methods with External Wearables: [7] proposed a mobile VR eye-tracking method, which can acquire eye images through a head-mounted camera without the assistance of additional equipment. [12] developed a head-mounted device to track line of sight and estimate user's gaze point. The head-mounted device can use in combination with a large endoscope camera, infrared light, and mobile phone. [10] built a system that can track high-speed eye movements, and the system can achieve a working frequency far 500 Hz. [16] proposed an automatic calibration method for 3D gaze estimation. With the development of deep learning, some scholars combine deep learning with eye trackers. [11] developed a pupil tracking technology based on deep learning for wearable device eye trackers.

Methods Without External Wearables: [17] designed a deep neural network architecture specifically for gaze estimation based on monocular input. [5] proposed a method to estimate people's overall attention in images. [6] took the relative position of key-points of the face as input for gaze estimation and used a confidence gated unit in the input layer of the network to deal with problems such as occlusion and incomplete key-points of the face. [3] proposed a face-based asymmetric regression evaluation network (FARE-Net). [4] proposed a model that simulates the dynamic interaction between scene and head features and infers the attention targets that change over time. [20] used a deep network structure to enable the robot to acquire human gaze or determine whether the user is making eye contact with the robot; still, it does not generate grasping and other interactive actions.

The above review shows that researchers have favored both gaze point estimation methods with external wearables and without external wearables, and the research on gaze point estimation has been increasing. Similarly, in HRI, researchers are keen to incorporate gaze points into HRI tasks to improve the naturalness of interaction. The following sections provide an overview of the usages of gaze points in HRI.

2.2 Application of Gaze Points in HRI

Modern-day approaches incorporating gaze points in human-robot interactions vary widely, and the completed tasks are also different. Based on the "GT3D" binocular eye tracker, [21] used Gaussian process regression to deal with the uncertainty of gaze point estimation to control the robotic arm better to complete the grasping task. [8] robot arm control by line-of-sight tracking so that the robotic arm can perform writing and painting artistic creations in 3D space. [23] developed a robot assistance system with intuitive and free gaze interaction, which estimates the user's gaze point in 3D space and controls the robotic arm for grasping and placing operations. [18] proposed a framework for grounding and learning examples through demonstration and eye-tracking (GLIDE). [9] built a platform based on a gaze tracking control robot, and users were able to navigate the mobile robot using only gazes as the system input. [24] proposed to use an automatic target location method combined with human gaze to extract relevant object positions, and this method can simplify the entire interaction process and improve the accuracy of interaction.

As seen from the above review, the methods to estimate gaze points can divide into those that rely on external wearables and those that do not. The method with external wearables is highly accurate but has the disadvantages of being inconvenient and costly to use. The way without external wearables uses deep learning techniques to enhance gaze point estimation accuracy greatly and is easy and inexpensive to use. In recent years, gaze points have also been used in a variety of HRI tasks. Still, most HRI tasks use external wearables (e.g., eye tracking devices) for estimation, which reduces the ease and naturalness of interaction and increases the cost of interaction. We combine the gaze point estimation method that does not rely on an external wearable and build an HRI framework that does not require the help of any external wearables and only requires RGB cameras.

3 Methods

This paper built a framework for human-robot interaction using only RGB cameras without the aid of other external wearables. This interaction framework requires inference of what object is gazed at by persons from global information, acquires the position of the object from local information in preparation for the mobile grabbing of the robot arm, entity matching through correspondence between global and regional information. In this framework, we, therefore, use the global view to obtain global information and use the local perspective to obtain local information.

3.1 Overview

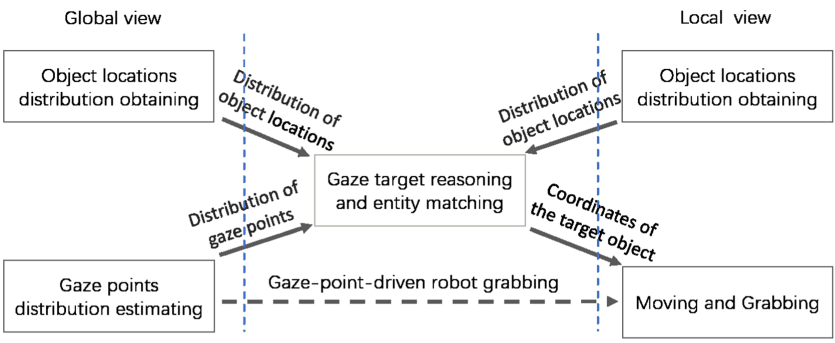

Fig. 1. Gaze-point-driven HRI framework.

As shown in Fig. 1, the global view mainly consists of object locations distribution obtaining and gaze points distribution estimating. The object locations distribution obtaining is used to obtain the position distribution of objects. The

gaze points distribution estimating is used to estimate the number of valid gaze points falling around each object. The gaze target reasoning and entity matching use gaze points distribution information from the global view to infer the gaze target and find the target object in the local view through a matching algorithm. The object locations distribution obtaining and the moving and grabbing are used in the local perspective. The object locations distribution obtaining has the same function as the object locations distribution obtaining module in the global view. The moving and grabbing convert the pixel coordinates of the objects in the local view into coordinates in the robot's coordinate system. It generates commands to drive the robot arm to move and grasp. The implementation details of the framework are described below.

3.2 Object Locations Distribution Obtaining

We use a vision-based object detection model to detect objects and determine the position distribution of each object, then filter and sort the detected object bounding boxes. Figure 2 is a flow chart of this module, showing the process of obtaining the distribution of object positions. The detailed implementations of duplicate border filters and reordering of border positions are described below.

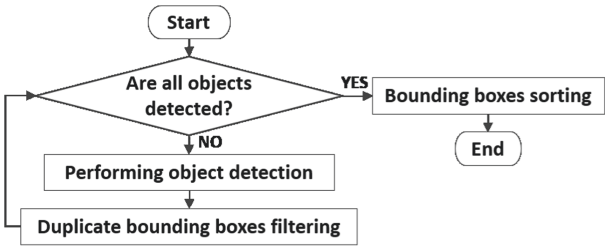

Fig. 2. Flowchart of the object locations distribution obtaining

Duplicate Bounding Boxes Filtering. In object detection, non-maximum suppression is used to determine the unique bounding box of an object. However, there is still the problem of duplicate bounding boxes for the same object during the experiments. Inspired by the non-maximum suppression algorithm, we solve this problem by suppressing the minimum distance between the center point of the object bounding boxes, as shown in filter f_d:

$$f_d(b_0) = \begin{cases} 0, \exists\ f_p(b_0, b_i) < \mathbb{N}, b_i \in s \\ 1, otherwise \end{cases} \tag{1}$$

where $s = \{b_1, b_2, b_3, \ldots, b_l | b_i =< (x_{min}, y_{min}), (x_{max}, y_{max}), w, h >\}, s\epsilon\ \mathbb{R}^{1\times l}$ is a storage unit that is used to save the object bounding boxes, l is the number

of saved object bounding boxes, $< (x_{min}, y_{min}), (x_{max}, y_{max}), w, h >$ denotes the top left coordinate, bottom right coordinate, width and height of the object bounding box, respectively. f_p is used to find the Euclidean distance between two points, b_0 is bounding box of the object to be saved currently, b_i is bounding box of the object saved in the s memory cell, and threshold $\mathbb{N}$ is the minimum distance. If f_d is 0, it means that there is a bounding box in s that overlaps with b_0 then b_0 is discarded. If it is 1, b_0 is saved to s.

Bounding Boxes Sorting. It means all objects have been detected when $f_n = 1$, but the position of the object bounding boxes saved in s may not correspond to the position of objects on the workbench. To ensure that the position order of the bounding boxes in the s corresponds to the object space position order, we use $sort(.)$ for sorting. The $sort(.)$ is a function that sorts objects according to the center point coordinates of their bounding boxes. Use f_n to verify that all objects have been detected, expressed as follows:

$$f_n(I) = \begin{cases} 1, if \ \sum_{b_i \in D(I)} f_d(b_i) = N \\ 0, otherwise \end{cases} \tag{2}$$

$$sort(s) = \{c(s_i) < c(s_j), i < j\} \tag{3}$$

where $D(.)$ is used for object detection and outputs the set of detected objects bounding boxes, $I \ \epsilon \ \mathbb{R}^{H \times W \times 3}$ image taken by the robot, $s \ \epsilon \ \mathbb{R}^{1 \times N}$is a storage unit that is used to save the object bounding boxes, N is the total number of objects, $c(s_i)$ denotes the center point coordinates of the i-th bounding box, When $i<j$, the coordinate value of the i-th bounding box is less than the coordinate value of the j-th bounding box.

3.3 Gaze Points Distribution Estimating

The distribution of valid gaze points around the object is obtained by first estimating the gaze points in the global view based on head information and scene information, processing the invalid gaze points, and recording the number of valid gaze points around each object. While estimating the gaze points, the robot also adjusts its interaction state based on the header information. Figure 3 is a flow chart of the gaze points distribution estimating, showing the process of obtaining a valid gaze points distribution, and the corresponding pseudocode is shown in Algorithm 1. The implementations of the robot's interaction mode adjustment and gaze point filtering are described in detail below.

Interaction Model Adjusting. In this framework, the robot may enter an invalid interaction mode for two reasons: 1) when a user with no interaction intent appears within the robot vision and wakes the robot into interaction mode, this is an invalid interaction mode. 2) when a user leaves during the interaction and does not have enough information about the gaze point to complete the subsequent interaction, the robot is in an invalid interaction mode.

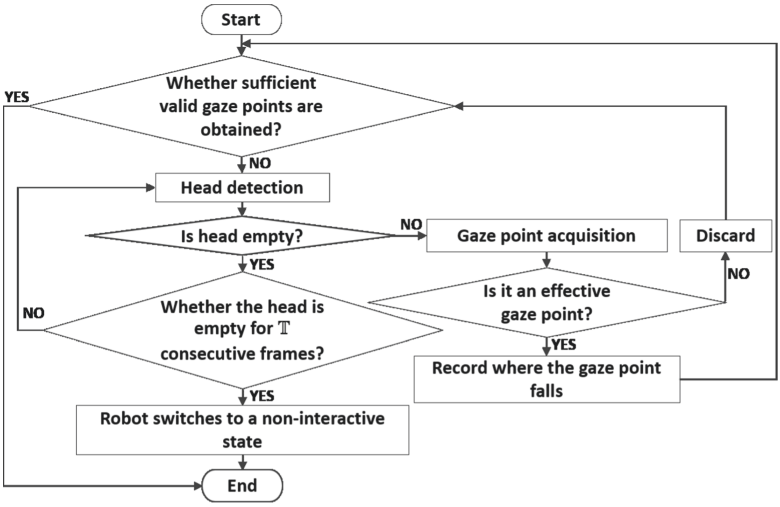

Fig. 3. Flowchart of the gaze points distribution estimating

The robot adjusts its mode according to the header information to avoid invalid interaction modes. If a person's head is not detected in $\mathbb{T}$ consecutive frames during the gaze point acquisition phase, $f_t = 0$, the robot will move from the interaction model to a non-interaction mode. f_t is an interaction status detector, which represents as follows:

$$f_{\Phi} = \begin{cases} 1, if\ \Phi(I) = 0 \\ 0, otherwise \end{cases} \tag{4}$$

$$f_t = \begin{cases} 0, if\ \sum_t^{t+\mathbb{T}} f_{\Phi} = \mathbb{T} \\ 1, otherwise \end{cases} \tag{5}$$

where t is the first time a video frame without head information is detected, $\Phi(.)$ is a head detector and returns the number of heads detected.

Gaze Points Filtering. To acquire the distribution of valid gaze points around each object, we start with determining whether the estimated gaze points are valid and filtering invalid gaze points, and finally, recording the distribution of valid gaze points around each object. This framework builds on the work of [4] to perform gaze point estimation based on the head pose.

Our filter focuses on two types of invalid gaze points in gaze point estimation: remote gaze points and saccade gaze points.

Remote Gaze Points: We call points far from the object above the workbench invalid remote gaze points, and we use maximum distance suppression to filter such points. We find the minimum distance of the gaze point from the nearest

Algorithm 1. Gaze points distribution estimating

input: s_g,s_n ▷ s_g is the storage unit to save the object bounding box in global view; s_n is used to record the number of valid gaze points g around each object, which is used to record the distribution of gaze points
output: s_n

$numtims \leftarrow 0$ ▷ Number of times the head was not detected
T ▷ Number of consecutive non-detects of a human head
$num \leftarrow 0$ ▷ Number of valid gaze points already detected
$Totalnum$ ▷ Total number of valid gaze points required
while $num<Totalnum$ **do**
  $Image \leftarrow image$ ▷ $image$: The image obtained from the global view
  $headBox \leftarrow headDetector(Image)$
  if $headBox \neq NULL$ **then** ▷ Detected head information
    $numtims \leftarrow 0$
    $gpoint \leftarrow GazePointEstimator(Image, headBox)$
    $index \leftarrow ValidityJudgment(gpoint, s_g)$
    if $index \neq 0$ **then** ▷ This gaze point is valid
      $s_n[index] \leftarrow s_n[index] + 1$
      $num \leftarrow num + 1$
    else ▷ This gaze point is invalid and discarded
      continue
    end if
  else ▷ Head information is not detected
    $numtims \leftarrow numtims + 1$
    if $numtims \neq T$ **then**
      continue
    else ▷ Head information is not detected in consecutive T-frames
      Robot switches to a non-interactive state
      return *null*
    end if
  end if
end while
return s_n

object and then apply maximum suppression of the minimum distance to filter the remote invalid gaze points. As indicated by f_{dis} :

$$f_g(g, s_g) = \begin{cases} 0, & if\ g \in b_i\ and\ b_i \in s_g \\ \min_{b_i \in s_g}(f_{ps}(g, b_i)), & otherwise \end{cases} \tag{6}$$

$$f_{dis}(g, s_g, s_n) = \begin{cases} 0, if\ f_g(g, s_g) > \mathbb{D} \\ 1, otherwise \end{cases} \tag{7}$$

where $s_g = \{b_1, b_2, b_3, \ldots, b_N | b_i =< (x_{min}, y_{min}), (x_{max}, y_{max}), w, h >\}$ is the storage unit used to save the object bounding box in global view, g is the gaze point (x, y), $s_n = \{n_1, n_2, n_3, \ldots, n_N | n_i \in \{0, 1, 2, 3, \ldots\}\}$ and s_g have the same size, and s_n is used to record the number of valid gaze points g around each object, indicating the number of times each object was gazed, N is the total

number of objects, b_i is the bounding box of the object saved in the s_g memory cell, n_i is the number of times the object at the corresponding position was gazed at, $< (x_{min}, y_{min}), (x_{max}, y_{max}), w, h >$ denotes the top left coordinate, bottom right coordinate, width and height of the object bounding box, respectively. f_{ps} is a function that calculates the shortest distance from point g to the rectangular region b_i formed by the bounding box, f_g is used to find the bounding box of the object in s_g that is closest to the gaze point g and the minimum distance, the threshold $\mathbb{D}$ is the maximum distance. If the distance returned by f_g is greater than $\mathbb{D}$, the gaze point g is invalid and the value of f_{dis} is 0, otherwise, f_{dis} is 1 and 1 is added to the position of s_n in relation to the nearest object, indicating that the corresponding object is gazed once, as illustrated in Fig. 4.

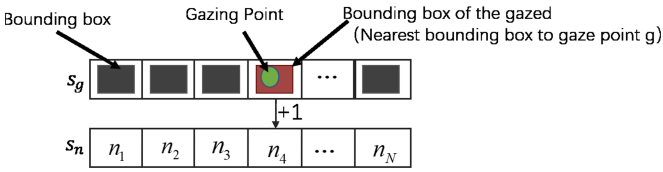

Fig. 4. The correspondence between s_g and s_n. where $n_1, n_2, n_3, ..., n_N$ denote the number of times the bounding box at the corresponding position has been gazed at. If the bounding box in s_g is gazed, 1 will be added to the position corresponding to it in s_n. Like n_4 will be added by 1.

Saccade Gaze Points: The gaze points detected during the user's movement are unstable gaze points, which we call saccade gaze points, and the small number of saccade gaze points cannot be used as a solid basis for reasoning about the gaze target. We solve the saccade gaze points problem by adopting a constraint on the minimum number of times that valid gaze points are detected, as denoted by f_s below:

$$f_s(g, s_g, s_n) = \begin{cases} 1, & if \ \sum_t^e f_{dis}(g, s_g, s_n) = \mathbb{Q} \\ -1, & otherwise \end{cases} \tag{8}$$

where the threshold $\mathbb{Q}$ is the total number of valid gaze points required, t is the starting frame, e is considered to be the end frame when $\mathbb{Q}$ valid gaze points are detected. Suppose the value of f_s is 1. In that case, it means that a sufficient number of valid gaze points have been obtained, and the gaze target can be inferred based on the distribution of the obtained valid gaze points; otherwise, the detection of valid gaze points will continue.

3.4 Gaze Target Reasoning and Entity Matching

The gaze target reasoning and entity matching module infers gaze targets based on the distribution of valid gaze points in the global view and object positions. It

uses a sorting-based entity matching method to match gaze targets in the local perspective. The implementations of gaze target reasoning and sorting-based entity matching will be described in detail below.

Gaze Target Reasoning. In this framework, we use an approach similar to human gaze reasoning, considering the location where the gaze point is most concentrated as the location where the gaze target is located. In Eq. 7, we can know the distribution of the valid gaze point g according to s_n. We consider the location of the maximum value in s_n as the location where the gaze points are most concentrated and consider this location as the location of the gaze target. With f_{ob}, we get the gaze target object in s_g based on this position. The expression is as follows:

$$f_{ob}(s_g, s_n) = f_v(s_g, f_m(s_n)) \tag{9}$$

$$f_v(s, index) = b_i, b_i \in s \; and \; i = index \tag{10}$$

where f_m is used to obtain the location of the gaze point concentration (the $index$ of the maximum value in s_n), f_v is a function for inferring the gaze target object from the storage cell based on the location of the gaze point concentration (Retrieve the object at $index$ position in the storage cell), $s = \{b_1, b_2, b_3, \ldots, b_N | b_i =< (x_{min}, y_{min}), (x_{max}, y_{max}), w, h >\}, s\epsilon \; \mathbb{R}^{1\times N}$ is a storage unit that is used to save the object bounding boxes, N is the number of saved object bounding boxes, b_i is the bounding box of the object saved in the s memory cell, $< (x_{min}, y_{min}), (x_{max}, y_{max}), w, h >$ denotes the top left coordinate, bottom right coordinate, width and height of the object bounding box, respectively.

Entity Matching. The entity matching method is a matching method that does not rely on object features but only on the order of object storage locations, which requires that the positions between objects in different views are relatively invariant and the position of the bounding boxes in the storage unit corresponds to each other. The expressions are as follows:

$$\Theta_g = \{\theta_1^g, \theta_2^g, \theta_3^g, \ldots, \theta_N^g | x(\theta_i^g) < x(\theta_j^g), i < j\} \tag{11}$$

$$\Theta_l = \{\theta_1^l, \theta_2^l, \theta_3^l, \ldots, \theta_N^l | x(\theta_i^l) < x(\theta_j^l), i < j\} \tag{12}$$

$$Match : \Theta_g \approx \Theta_l \tag{13}$$

where both θ_i^g and θ_i^l are composed of $< (x_{min}, y_{min}), (x_{max}, y_{max}), w, h >$, Θ_g and Θ_l are ordered sets that store the bounding boxes of objects in the global and local views. $x(\theta_i^g)$ and $x(\theta_i^l)$ denote the x-value of the coordinates of the centre point of the i-th bounding box in the global view and local view, respectively. When $i<j$, the x-value of the coordinate of the centre point of the i-th bounding box is less than the x-value of the coordinate of the centre point of the j-th bounding box. N is the number of entities, $< (x_{min}, y_{min}), (x_{max}, y_{max}), w, h >$

denotes the top left coordinate, bottom right coordinate, width and height of the object bounding box, respectively. In the matching process, we consider that the same location of Θ_g and Θ_l stores the bounding box of the same object under different views. Figure 5 provides a more intuitive representation of the sorting-based entity matching method.

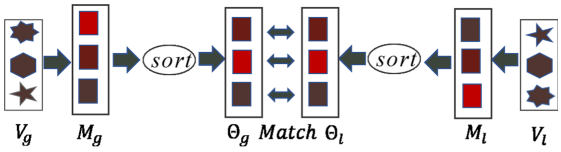

Fig. 5. Sort-based entity matching: V_g, V_l are the object positions in the global and local views, and M_g and M_l store the bounding boxes of the objects in the global and local views, respectively, which are sorted by *sort* to obtain the ordered sets Θ_g and Θ_l. The bounding boxes of the same object are stored in the same order in both Θ_g and Θ_l, so that during the matching process, we assume that Θ_g and Θ_l store the same object at the same location. *sort* is a sorting function, like Eq. 3.

3.5 Moving and Grabbing

The moving and grabbing module focuses on moving the arm to the target object and grabbing it. Figure 6 shows the flow of the operation. Converting the pixel coordinates of the target object to world coordinates in the Baxter robot's coordinate system is the key to the moving procedure, and we will describe the coordinate conversion in more detail below.

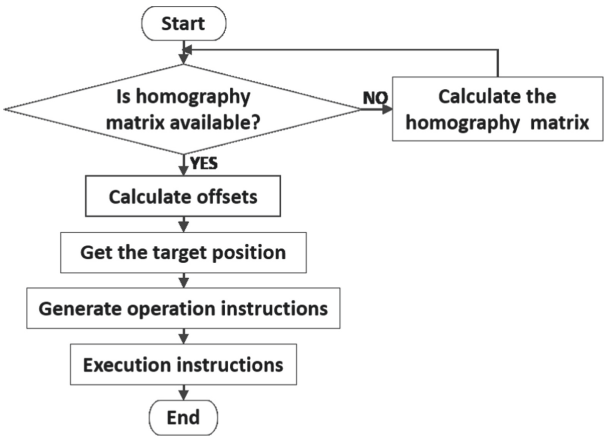

Fig. 6. Flowchart of the moving and grabbing

Coordinate Conversion. The coordinate conversion is primarily used to convert pixel coordinates to robot coordinates. The pixel position information of the bounding box of the target object obtained using Eq. 9 is in the pixel coordinate system. It cannot be used for the control of the robot directly. We use the calibration board to calculate the homography matrix from the camera image plane to the working plane. The homography matrix is used to find the offset between the center point of the object bounding box and the visual center point in the robot's base coordinate system. The target position is obtained based on this offset and the current position of the robot's end. The expressions show below:

$$P = P_r + H \times (C_{ob} - C_v) + \gamma \tag{14}$$

where $H \epsilon \mathbb{R}^{3\times3}$ is the homography matrix. $C_{ob} \epsilon \mathbb{R}^{3\times1}$, $C_v \epsilon \mathbb{R}^{3\times1}$ are the homogeneous coordinates of the object bounding box centre point and the visual centre point, respectively. P_r is the current position at the end of the robot arm, γ is the deviation compensation, P is the target position at the end of the robot arm.

4 Results

Experiments were conducted using the Baxter robot under different lighting conditions, different interaction distances, and different persons to verify the robustness of the interaction framework. In Subsect. 4.1, we will describe our experimental equipment and experimental setup. In Subsect. 4.2, we will analyze the experimental results. Presentation videos are listed as following: daytime [Youku] or [Youtube], nighttime [Youku] or [Youtube].

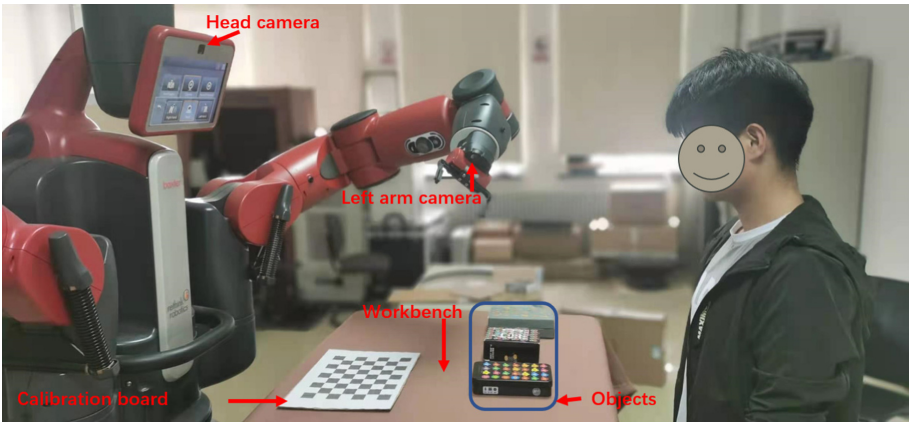

Fig. 7. Experimental scene. Including a calibration board, a workbench, the object to be grasped, and the Baxter robot head and left arm camera

4.1 Experimental Equipment and Setup

Throughout the experiment, we used a computer with an RTX2080 GPU, a workbench, a calibration board, the object to be grasped, a Baxter robot with parallel grippers, and the Baxter robot head camera and left arm camera as hardware devices, as shown in Fig. 7. The Baxter head camera is a global view camera, and the Baxter left arm camera is a local view camera. We implement the program using a mixture of python 3.7 and the C++ programming language.

In this experiment, we need the user's head information, and we trained Yolov4 [2] as a head detector based on the head dataset provided in [19]. During the training process, we set max_batches to 5000 depending on the amount of data and the number of categories. We performed the training on a computer with an RTX2060 and achieved an mAP of 95.5%. In this interaction framework, we also use Yolov4 trained on the Microsoft coco [14] dataset as an object detector. The separation distance between objects on the workbench is controlled to be around 10 cm.

As shown in Fig. 8, we set the interaction distance (the distance of the experimenter from the workbench) into three distance bands: near (within 60 cm), medium (60 cm–120 cm), and far (120 cm–200 cm), and calculate the mission success rate for different distances.

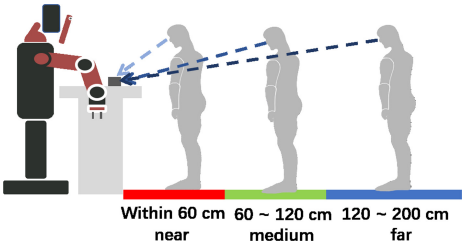

Fig. 8. Interaction distance. Including three distance bands: near, medium, and far

4.2 Experimental Results

To verify the robustness of this interaction framework under different lighting conditions, we conduct experiments under natural daylight exposure and evening lighting conditions. The data in Table 1 shows that the average success rates are 90.0% and 91.1% for near range, 87.8% and 82.2% for medium range, 76.7% and 74.4% for far range under natural daylight and evening lighting conditions, respectively. Although there is a difference in the average success rate of this interaction framework under different lighting conditions, the difference is not significant. The average success rate for each person in daytime natural lighting conditions is also similar to their average success rate in evening lighting conditions. This all proves that this interaction framework is robust under different lighting conditions.

Table 1. Task success rate and average success rate of different experimenters under different illumination and different interaction distance conditions. U_*^c (this experimenter wears glasses), $*\_f$ (this experimenter is a female), *_N (near), *_M (medium), *_F (far), Nl_* (natural daylight exposure conditions), El_* (evening lighting conditions). The data format for each row is several successes/total number of executions (s/t). AVGSR1 (the average success rate of each experimenter under natural lighting conditions during the day and night lighting conditions), AVGSR2 (the average success rate of different interaction distances under different lighting conditions). 83.7% (the overall average success rate).

Experimenters	Nl_N	Nl_M	Nl_F	El_N	El_M	El_F	AVGSR1
U_1	10/10	9/10	10/10	9/10	9/10	9/10	93.3%
U_2^c	8/10	10/10	8/10	9/10	8/10	8/10	85.0%
U_{3_f}	10/10	9/10	8/10	10/10	10/10	7/10	90.0%
U_4^c	9/10	10/10	9/10	10/10	8/10	8/10	90.0%
U_{5_f}	9/10	7/10	8/10	9/10	8/10	6/10	78.3%
U_6^c	9/10	8/10	5/10	9/10	6/10	7/10	73.3%
U_7^c	8/10	9/10	9/10	8/10	7/10	8/10	81.7%
$U_{8\_f}^c$	9/10	9/10	5/10	10/10	9/10	8/10	83.3%
U_{9_f}	9/10	8/10	7/10	8/10	9/10	6/10	78.3%
AVGSR2	90.0%	87.8%	76.7%	91.1%	82.2%	74.4%	**83.7%**

The AVGSR2 data in Table 1 shows that, under natural daylight conditions, the average success rates at a near, medium, and far distance are 90.0%, 87.8%, and 76.7%, respectively; under evening lighting conditions, are 91.9%, 82.2% and 74.4%, respectively. It is clear from the data that the average success rate decreases as the distance gets further. The individual data for each person also shows that the success rate decreases as the distance gets further. Although the average success rate is gradually decreasing, the average success rate exceeds 90.0%, 82.2%, 74.4% for near, medium, and far distances, respectively, which shows that this interaction framework is robust and effective at different interaction distances.

To verify the robustness of this interactive framework across different body types, heights, clothing, genders, and in the presence of other glasses. Nine experimenters (five male and four female, different body types, heights, clothing, and some wearing glasses) are invited to perform the same interactive task under natural daylight exposure and evening lighting conditions. To exclude the effects of light and interaction distance, we analyze the data in Table 1 under the same light and interaction distance conditions. There is little difference in the number of successes between the near and medium-distance person. The number of wins in the far distance interactions is not significantly different, except for a person's relatively low number of successes. In Table 1, the average success rate is 84.66% for males and 82.48% for females, with a slight difference between the two percentages. The average success rate is 82.66% for those wearing glasses

and 84.98% for those not wearing glasses, with a bit of difference between the two percentages. The data suggest that this interaction framework is robust across different body types, heights, clothing, and gender conditions, and that person can still interact efficiently when wearing other glasses.

The above experimental results show that this interaction framework is highly robust and can successfully perform interaction tasks at different interaction distances, in different lighting environments, and with users of various sizes. Wearing other glasses does not affect the efficient use of this interaction framework. However, this interaction framework can only be used in a single-person situation, which is a limitation of this framework and a problem we will address in the future.

5 Conclusion

In this work, the human-robot interaction framework using only RGB cameras without the assistance of any external wearables is built. In this framework, the robot uses only RGB cameras to infer the target of the person's gaze and grasps it. The interaction framework does not require external wearables, makes interaction more accessible and reduces interaction costs. We conducted several sets of experiments based on the Baxter robot and demonstrated the high robustness of our interaction framework.

As this framework uses only RGB cameras during the interaction, a calibration board is required to assist in the conversion of coordinates. The entity matching algorithm converts the target object from the global view to the local perspective to convert coordinates more accurately, resulting in a longer interaction time. In future work, we will focus on solving the conversion problem and implementing and extending the framework to accommodate multi-player human-robot interaction.

Acknowledgment. This work was supported in part by the Key Program of NSFC (Grant No. U1908214), Special Project of Central Government Guiding Local Science and Technology Development (Grant No. 2021JH6/10500140), Program for the Liaoning Distinguished Professor, Program for Innovative Research Team in University of Liaoning Province, Dalian and Dalian University, the Scientific Research fund of Liaoning Provincial Education Department (No. L2019606), Dalian University Scientific Research Platform project, and in part by the Science and Technology Innovation Fund of Dalian (Grant No. 2020JJ25CY001).

References

1. Acien, A., Morales, A., Vera-Rodriguez, R., Fierrez, J.: Smartphone sensors for modeling human-computer interaction: general outlook and research datasets for user authentication. In: 2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC), pp. 1273–1278. IEEE Computer Society, Los Alamitos (2020)

2. Bochkovskiy, A., Wang, C.Y., Liao, H.Y.M.: Yolov4: optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934 (2020)
3. Cheng, Y., Zhang, X., Lu, F., Sato, Y.: Gaze estimation by exploring two-eye asymmetry. IEEE Trans. Image Process. **29**, 5259–5272 (2020)
4. Chong, E., Wang, Y., Ruiz, N., Rehg, J.M.: Detecting attended visual targets in video. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5395–5405. IEEE Computer Society, Los Alamitos (2020)
5. Chong, E., Ruiz, N., Wang, Y., Zhang, Y., Rozga, A., Rehg, J.M.: Connecting Gaze, scene, and attention: generalized attention estimation via joint modeling of gaze and scene saliency. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11209, pp. 397–412. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01228-1_24
6. Dias, P.A., Malafronte, D., Medeiros, H., Odone, F.: Gaze estimation for assisted living environments. In: 2020 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 279–288. IEEE, Snowmass Village, Colorado (2020)
7. Drakopoulos, P., Koulieris, G.A., Mania, K.: Front camera eye tracking for mobile VR. In: 2020 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW), pp. 642–643. Atlanta (2020)
8. Dziemian, S., Abbott, W.W., Faisal, A.A.: Gaze-based teleprosthetic enables intuitive continuous control of complex robot arm use: writing & drawing. In: 2016 6th IEEE International Conference on Biomedical Robotics and Biomechatronics (BioRob), pp. 1277–1282. IEEE, University Town, Singapore (2016)
9. Gêgo, D., Carreto, C., Figueiredo, L.: Teleoperation of a mobile robot based on eye-gaze tracking. In: 2017 12th Iberian Conference on Information Systems and Technologies (CISTI), pp. 1–6. IEEE, Lisbon, Portugal (2017)
10. Hosp, B., Eivazi, S., Maurer, M., Fuhl, W., Geisler, D., Kasneci, E.: Remoteeye: an open-source high-speed remote eye tracker:implementation insights of a pupil- and glint-detection algorithm for high-speed remote eye tracking. Behav. Res. Methods **52**(3), 1387–1401 (2020)
11. Kuo, T.L., Fan, C.P.: Design and implementation of deep learning based pupil tracking technology for application of visible-light wearable eye tracker. In: 2020 IEEE International Conference on Consumer Electronics (ICCE), pp. 1–2. IEEE, Las Vegas (2020)
12. Lee, K.F., Chen, Y.L., Yu, C.W., Chin, K.Y., Wu, C.H.: Gaze tracking and point estimation using low-cost head-mounted devices. Sensors **20**(7) (2020)
13. Li, X.: Human-robot interaction based on gesture and movement recognition. Sig. Process. Image Commun. **81**, 115686 (2020)
14. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
15. Liu, J., Chang, W., Li, J., Wang, J.: Design and implementation of human-computer interaction intelligent system based on speech control. Comput.-Aid. Des. Appl. **17**, 22–34 (2020)
16. Liu, M., Fu Li, Y., Liu, H.: 3D gaze estimation for head-mounted devices based on visual saliency. In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 10611–10616. IEEE, Las Vegas (2020)
17. Park, S., Spurr, A., Hilliges, O.: Deep pictorial gaze estimation. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11217, pp. 741–757. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01261-8_44

18. Penkov, S., Bordallo, A., Ramamoorthy, S.: Physical symbol grounding and instance learning through demonstration and eye tracking. In: 2017 IEEE International Conference on Robotics and Automation (ICRA), pp. 5921–5928. IEEE, Marina Bay Sands (2017)
19. Radhakrishnan, P.: Head-detection-using-yolo. https://github.com/pranoyr/head-detection-using-yolo. Accessed 4 Oct 2020
20. Saran, A., Majumdar, S., Short, E.S., Thomaz, A., Niekum, S.: Human gaze following for human-robot interaction. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 8615–8621. IEEE, Madrid (2018)
21. Tostado, P.M., Abbott, W.W., Faisal, A.A.: 3D gaze cursor: continuous calibration and end-point grasp control of robotic actuators. In: 2016 IEEE International Conference on Robotics and Automation (ICRA), pp. 3295–3300. IEEE, Stockholm (2016)
22. Tsai, T.H., Huang, C.C., Zhang, K.L.: Design of hand gesture recognition system for human-computer interaction. Multimedia Tools Appl. **79**(9), 5989–6007 (2020)
23. Wang, M.Y., Kogkas, A.A., Darzi, A., Mylonas, G.P.: Free-view, 3D gaze-guided, assistive robotic system for activities of daily living. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2355–2361. IEEE, Madrid (2018)
24. Weber, D., Santini, T., Zell, A., Kasneci, E.: Distilling location proposals of unknown objects through gaze information for human-robot interaction. In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 11086–11093. IEEE, Las Vegas (2020)