Gesture and Vision Based Automatic Grasping and Flexible Placement in Teleoperation

zhao xue (876421149@qq.com)
Chongqing University https://orcid.org/0000-0003-4911-1900
CHEN Xiaoan
Chongqing University
HE Ye
Chongqing University
Hongli Cao
Chongqing University
Tian Shengli
Chongqing University

Research Article

Keywords: Gesture, Grab, human-robot interaction, telerobot, vision Declarations

Posted Date: May 17th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-510392/v1

Version of Record: A version of this preprint was published at The International Journal of Advanced Manufacturing Technology on February 2nd, 2022. See the published version at https://doi.org/10.1007/s00170-021-08585-z.

Title page

Title

Gesture and vision based automatic grasping and flexible placement in teleoperation

Author information:

Zhao Xue

State Key Laboratory of Mechanical Transmission, Chongqing University, Chongqing 400044, China.

Xiaoan Chen

State Key Laboratory of Mechanical Transmission, Chongqing University, Chongqing 400044, China.

Ye He

State Key Laboratory of Mechanical Transmission, Chongqing University, Chongqing 400044, China.

Hongli Cao

State Key Laboratory of Mechanical Transmission, Chongqing University, Chongqing 400044, China.

Shengli Tian

1. School of Mechantronics and Vehicle Engineering, Chongqing Jiaotong University, Chongqing 400074, China;
2. Chongqing Changjiang Bearing Co., Ltd., Chongqing 401336, China;
3. Chongqing Key Laboratory of Manufacturing Equipment Mechanism Design and Control, Chongqing Technology and Business University, Chongqing 400067, China

Corresponding author: Ye He

Corresponding Author's Email: hifish2@gmail.com

Abstract

Teleoperation system has attracted a lot of attention because of its advantages in dangerous or unknown environment. It is very difficult to develop an operating system that can complete complex tasks in an completely autonomous. This paper proposes a robot arm control strategy based on gesture and visual perception. The strategy combines the advantages of humans and robots to obtain a convenient and flexible interaction model. The hand data were obtained by Leap-Motion. Then a neural network algorithm was used to classify the nine gestures used for robot control by a finite state machine. The control mode switched between indicative control and mapping control. The robot acquired a autonomous grasp ability by incorporating YOLO 6D, depth data, and a probabilistic roadmap planner algorithm. The robot completed most of the trajectory independently, and a few flexible trajectories required a user to make mapping actions. This interactive mode reduces the burden of the user to a certain extent, that makes up for the shortcomings of traditional teleoperation.

Keywords

Gesture, Grab, human-robot interaction, telerobot, vision Declarations

Declarations

Funding

The project is supported by The National Key Research and Development Program of China（No.2017YFB1301400), Industrial verification platform and performance evaluation of precision machine tool spindle bearing of The National Key Research and Development Program of China（No. 2018YFB2000500) and Graduate Scientific Research and Innovation of Chongqing (NO. CYB19062).

Conflicting of interest

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Code availability

Not applicable

Availability of data and material

All data generated or analysed during this study are included in this published article.

Ethics approval

Not applicable

Consent to participate

Written informed consent to participate was obtained from all participants.

Consent for publication

Written informed consent for publication was obtained from all participants.

Gesture and vision based automatic grasping and flexible placement in teleoperation

Zao Xue[1], Xiaoan Chen[1], Ye He[1]*, Hongli Cao[1], Shengli Tian[2]

(1.State Key Laboratory of Mechanical Transmission, Chongqing University, Chongqing 400044, China.
2. Chongqing Key Laboratory of Manufacturing Equipment Mechanism Design and Control, Chongqing Technology and Business University, Chongqing 400067, China)

ABSTRACT: Teleoperation system has attracted a lot of attention because of its advantages in dangerous or unknown environment. It is very difficult to develop an operating system that can complete complex tasks in an completely autonomous. This paper proposes a robot arm control strategy based on gesture and visual perception. The strategy combines the advantages of humans and robots to obtain a convenient and flexible interaction model. The hand data were obtained by Leap-Motion. Then a neural network algorithm was used to classify the nine gestures used for robot control by a finite state machine. The control mode switched between indicative control and mapping control. The robot acquired a autonomous grasp ability by incorporating YOLO 6D, depth data, and a probabilistic roadmap planner algorithm. The robot completed most of the trajectory independently, and a few flexible trajectories required a user to make mapping actions. This interactive mode reduces the burden of the user to a certain extent, that makes up for the shortcomings of traditional teleoperation.

Keywords: Gesture, Grab, human-robot interaction, telerobot, vision

1. Introduction

With the development of computer vision, the tasks that autonomous robots can accomplish become more complex. However, in the highly unstructured dynamic environment, the object is unfamiliar, the shape changes, or the motion is unknown. For complex tasks, the decision-making and control of robots need human intelligence, especially in complex and dangerous environments. When a human does not want or is difficult to appear in the robot field, it is necessary to remotely operate the robot. Such cases include nuclear energy [1], ocean exploration [2], and many others.

The commonly used human-robot interface for remote operation includes various equipment, like exoskeleton equipment [3], electromagnetic [4] or motion capture devices[5], or EMG devices [6]. However, as these devices are attached to the human body, they may affect the comfort and flexibility of human operation. Other controlling devices, such as actuator replicas, control panel, joystick, and mouse [7], need that the operator to learn extra operation skills. Gesture interaction based on non-contact [8] has been studied and integrated with speech recognition to give users a more intuitive and natural interactive experience. However, these interface modes are relatively direct to deal with some small tasks, such as moving, rotating, running and stopping, etc. These

modes would be difficult to deal with complex and large-scale actions. The advantages of combing humans and robots are not considered the task can segment according to the areas where robots and humans are good at. Currently, the development direction is intelligent; improving the interaction efficiency and reducing the work intensity and the cost. Also, some slight indicative actions can trigger the robot to complete certain tasks through eye contact and brain signals [9], which reduces the human's work effort.

In addition to the model of direct control[10, 11], supervision is another way to reduce working strength. Supervised autonomy is one model that allows the operator to start tasks while the execution is autonomously [12, 13]. It allows the operator to control the robot without concentration, thus reducing the task intensity of the operator. On the other hand, in sharing or mapping control, continuous input on the human-computer interface is used to control robot [12], which increases human input, such as protection function and automatic collision avoidance. Hayati [14] provides strategies in robot systems, including supervised autonomy and shared control. Bauer [15] combines the two models of operation via semantics. Semantic is easy to implement for simple commands, such as moving 10cm to the left while it is difficult for complex tasks, such as following a trajectory. From the perspective of spatial scope, the recognition range of the hand is limited. In other words, the hand is not suitable for moving in a large range. Therefore, Du [16] intentionally made a 3-dimensional mobile platform to increase the mobile range of gesture recognition. Accordingly, we can enable the supervision model to perform a wide range of movement, and direct control mode in the small range of flexible movement may also able to solve this problem. With regard to obstacle avoidance, the author uses the Neural-Learning-Based method to avoid obstacles [17], which can control each joint. To simplify obstacle avoidance and improve operation efficiency, we use a random node generation method to avoid obstacles that only considers the end of the robot [18].

Grasping is a normal behavior for humans, but it consists of considerable complex cognitive and biomechanical processes [19]. Also, for many innovative robot systems, grasping and manipulating objects are necessary. Robot grasping technology is also an important technology for the next generation industry. In 2016, in the Amazon grabbing challenge, the team who used the fast R-CNN algorithm won the champion of grabbing and stacking projects [20]. Sahin [21] classifies object pose recognition into five categories, which are based on classification, regression, classification and regression, template matching, and point cloud feature matching. To improve the recognition speed, the YOLO [22] formulates the detection problem as a regression problem and directly predicts the bounding box and classification probability from the image.However, the image recognition should not be only limited in the 2D plane but needs to know the position and posture of the object. Based on this, Tekin [23] designed YOLO 6D target attitude prediction method, which

has high recognition accuracy in complex scene attitude estimation. In this study, indicative actions and mapping gestures were incorporated to reduce the effort of user engagement. The HRC model combined object recognition, gesture recognition, and a finite state machine (FSM). Automatic teleoperation grasping and placing mode was devised, and object verification was done in a real environment. The robot identified the object and planned the trajectory and movement by itself. The user gave some demand hints so that the robot could grab the appropriate objects. On the other hand, the robot could be seamlessly switched to a mapping mode to place the object, which enabled good flexibility.

Special contributions of the research include the following:

1. Leap-Motion (Ultraleap Holdings, Ltd., UK) was used to acquire human finger gestures. The gestures were related to the two interactive modes of indicating and human-robot mapping, which combined with the FSM to complete the tasks of grasping, obstacle avoidance, and placing.
2. The object detection architecture YOLO 6D's attitude recognition, based on a 2D plane, was applied to a real robot grasp. The output was combined with point cloud data to meet the grasping requirements.

2. The interaction system structure

As shown in Fig. 1, the physical structure of the system was composed mainly of a mechanical arm, a gesture recognition device, and visual recognition. The robot arm manipulated objects with its gripper. The gesture recognition device recognized user gestures and controlled the operation of the manipulator. The visual system displayed scene information, including object information and scene depth. Two cameras were used, one for the front view and the other for the side view, to help the operator control the robot.

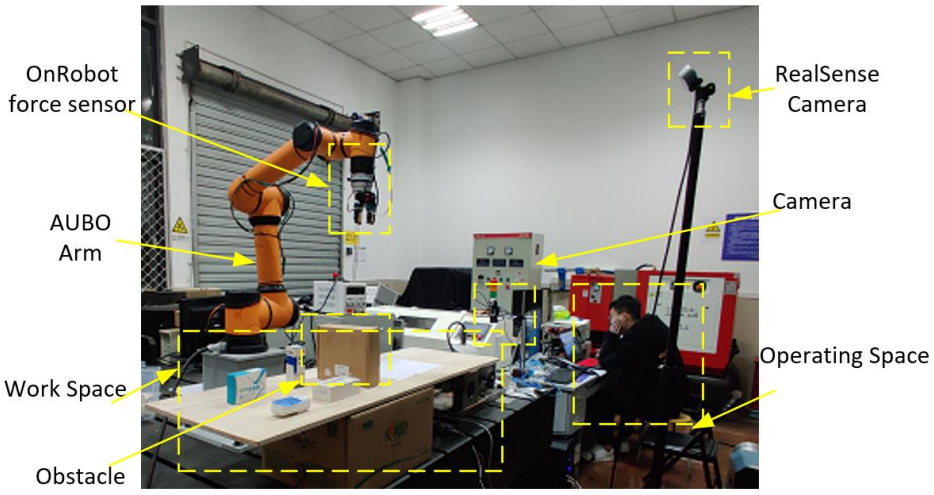


Figure 1. Scene of human-robot interaction

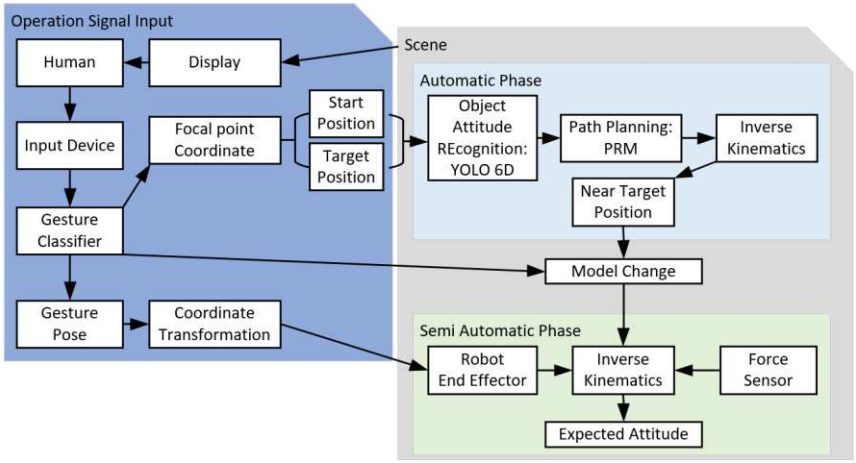


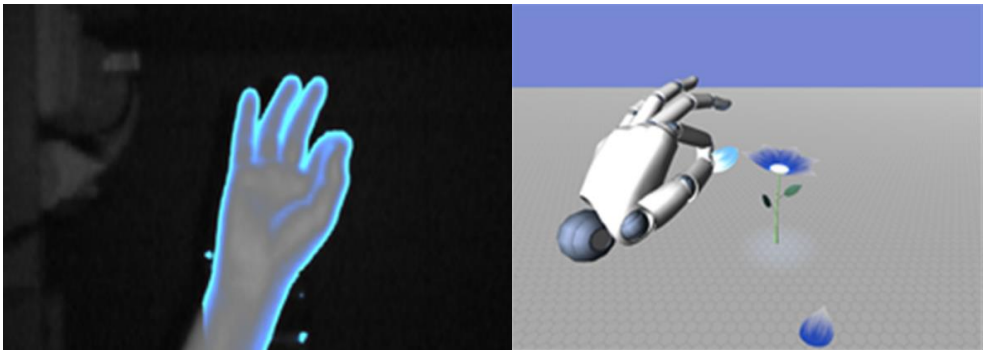
Figure 2. Interaction system structure

Figure 2 shows the data structure of the system, which comprised two main parts: Part 1 collected human data and translated them to the scene. In Part 2, the robot got data from humans and the environment, grasped automatic ally and positioned the object based on a human gesture pose. In Fig. 2, the blue box on the left represents the user. Hand data were collected through images and transformed into a state and a position. The position information was based on the user's hand indication points, which could be recognized by the computer through a series of algorithms. The grey box on the right is the data flow of the system, which involved mainly object recognition, obstacle avoidance, motion planning, kinematics, and inverse kinematics, and coordinate transformation. The object pose recognition was based on YOLO 6D imaging, which was written in Python. The obstacle avoidance algorithm was based on an improved probabilistic roadmap planner (PRM). The state switching was based on an FSM from MATLAB's Simulink.

2.1 Human gesture data collection

2.1.1 Input method

For convenience, the gesture was chosen as the input method. Leap-Motion is small and positioned on a desktop approximately 20 cm away from the hand, or positioned on the head in combination with VR glasses to input human gestures [24]. As shown in Fig. 3a, Leap-Motion can identify hand postures well and has a strong ability to avoid interference from the environment due to its mode, which is based on infrared light. Leap-Motion combined with Unity can do some beautiful and subtlehuman–computer interactions, such as grasping the petal of a flower (Fig. 3b).

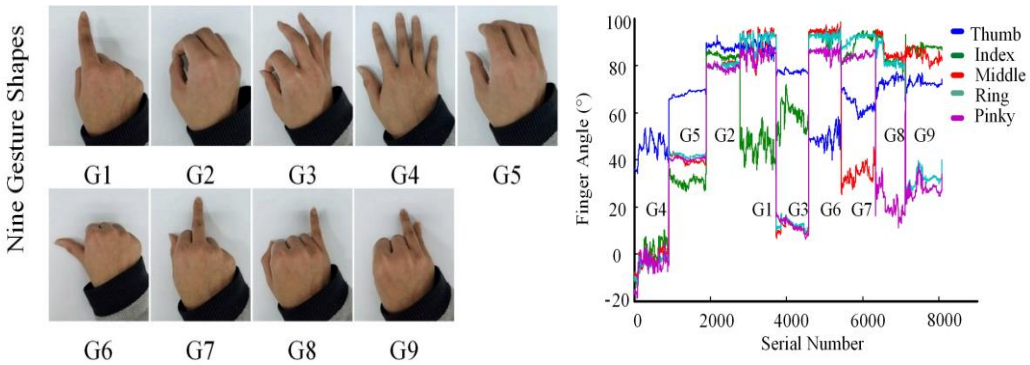


(a) (b)

Figure 3. Immersive interaction with (a) Leap-Motion and (b) Unity

2.1.2 Gesture classifier

To obtain gesture transformation, a set of typical gestures was used (Fig. 4a), based on previous research[25, 26]. The corresponding angle information of the fingers is shown in Fig. 4b. The network structure is shown in Fig. 5. w1, w2 and w3, mean weight. b1, b2, and b3 mean bias.



(a) (b)

Figure 4. (a) Gesture classifications. (b) G1 to G9 correspond to the bending angles of the fingers.

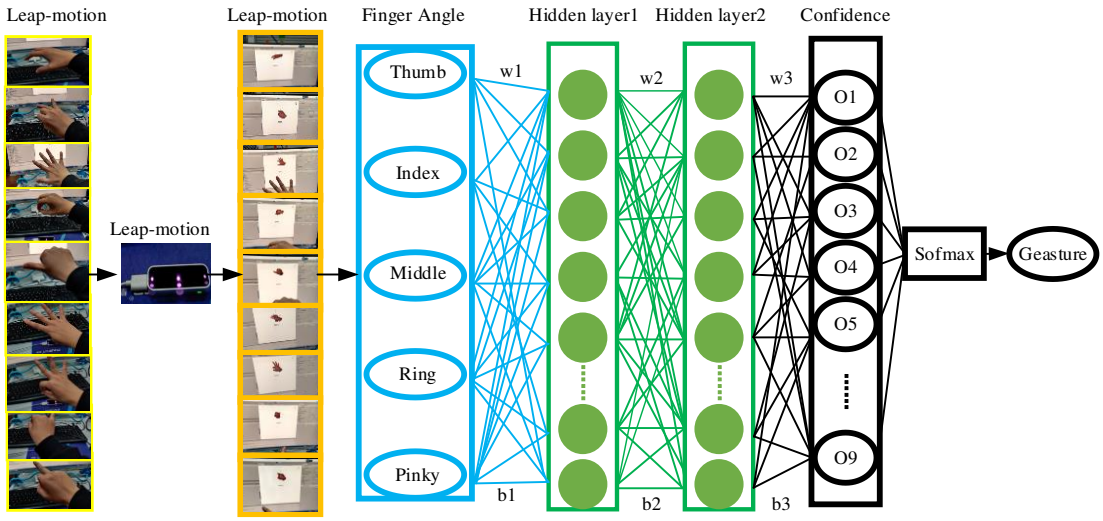


Figure 5. Structure of the finger recognition neural network

The input-to-output hidden layer takes the H 'th row of W and multiplies that row by x:

$$H_i = \sum_{i=1}^{d} W_{yi} x_i + b_i . \tag{1}$$

From hidden layer to output layer:

$$O_i = \sum_{i=1}^{d} W_{yi} x_i + b_i . \tag{2}$$

Sigmoid is used to add nonlinear factors:

$$S(x) = \frac{1}{1+\exp(-x)} . \tag{3}$$

The softmax function is applied to get normalized probability:

$$P(G \mid x) = \frac{\exp(W_G \cdot x)}{\sum_{c=1}^{C} \exp(W_c \cdot x)} . \tag{4}$$

2.1.3 Indicative point of gestures focal

The position and indication vector of the finger could solve the focal point to the plane only when the gesture was G1. P_i is the position of the index finger, L is the direction, P_o is the intersection point between the straight line and the plane, N is the normal vector of the plane, and D is the distance to the plane. The point P_o canbe calculated as shown in Eq. (5) and (6).

$$P_o = P_i + DL \tag{5}$$

$$\cdot (P_o - P_i) N = 0 \tag{6}$$

Select the object closest to point P_o as the target object for attitude display: O is the object

coordinate and j is the subscript:

$$\text{Object} = \arg\min_{\text{j}} \left\| P_O - O_{\text{j}} \right\|^2 \tag{7}$$

2.2 Robot automatic grasping

2.2.1 Scene generation

For scene modeling, positioning, and navigation, a depth camera such as an RGB-D [27] or LIDAR [28] is commonly used. The advantage[29] of the RGB-D is that it can obtain the scene information with good performance at a relatively low price and can display a colour image and depth data. Commonly used equipment such as RealSense and Kinect [30] are suitable for indoor and other small scenes. The RealSense second-generation D415 with a range of 0.3 to 10 m. Typically, the D415 can capture colour and depth images concurrently, with a high frame rate of 60 frames per second. A single frame comprises a colour image and a depth image and contains geometrical 3D information. The generated depth data can be converted to the point cloud.

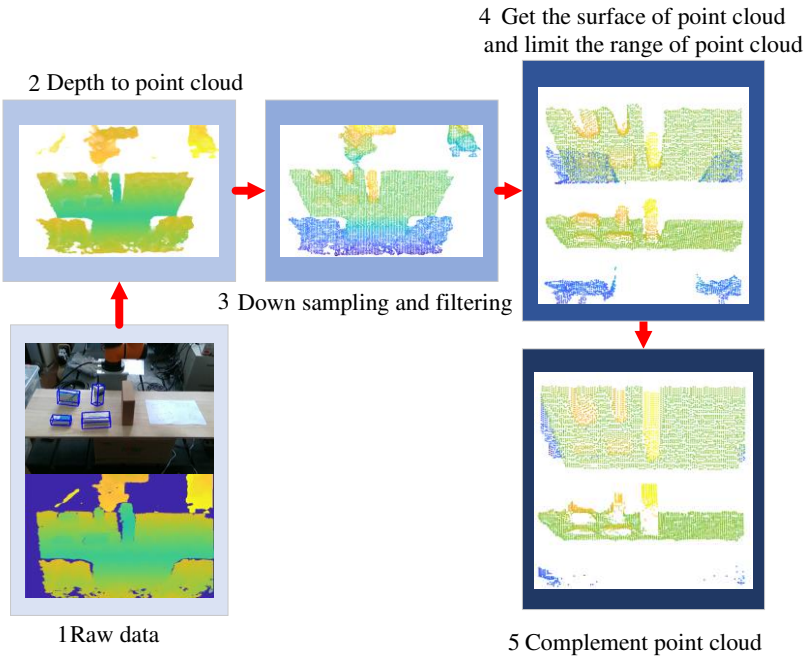


Figure 6. Scene generation using a depth camera

Depth data cannot be used directly, but requires a series of data processing, as shown in Fig. 6. Depth data that determines the distance between the object and the camera mapped to the pixel coordinate system do not directly reflect the Cartesian coordinates of the object in the scene. Also, the depth camera is subject to environmental interference that may cause noise, and a large amount of depth information data will affect the real-time performance of data interaction. To prevent this, the first step (1 in Fig. 6) was to turn the deep data into a point cloud and detect the presence of

noise. Due to the very large amount of data, direct filtering would consume many computing resources, so step 2 was to subsample and then filter. If the camera was tilted, part of the area would inevitably be lost after rotation, so step 3 was done to limit the range of the point cloud and obtain the local highest point. Step 4 was done to make up for the point cloud by combining the highest point cloud and the threshold value and expanding outward with the camera position as the starting position.

2.2.2 Object attitude recognition and trajectory planning

This Object attitude recognition is based on the YOLO 6D[23], and the network structure is in Fig. 7.

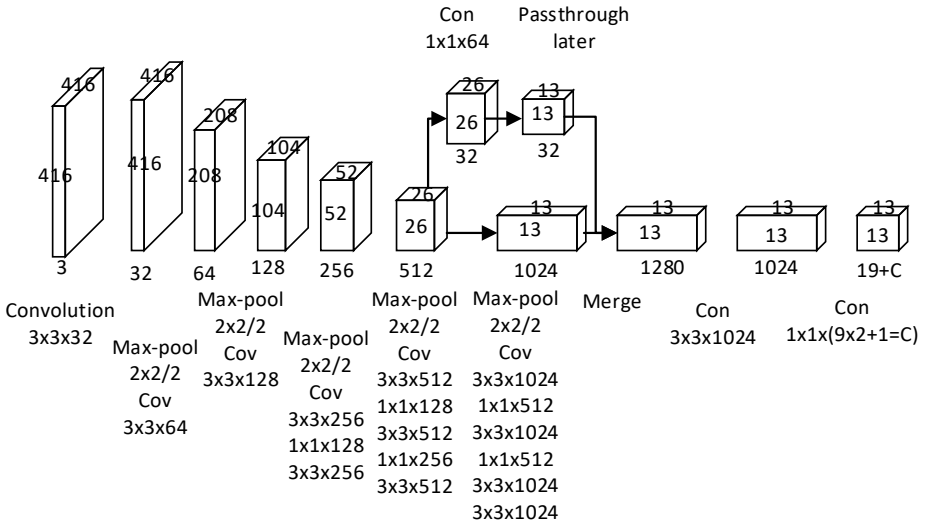


Figure 7. YOLO 6D network structure

In the training [27] and use of models, YOLO 6D converted the picture to 416 × 416. The final output of 13 × 13 × (9 × 2 + 1 + C) indicated the output of two boxes, with nine points in each box; one confidence value; and C class possibilities. The network had to predict the coordinates of nine points, including eight corner points and one center point. However, it did not predict the coordinate value directly: just like YOLO V2, it predicted the offset relative to the cell [Eq. (8)], where c_x and c_y were the coordinates of the cell. For the center point, $f(.)$ was the sigmoid function, and for the corner point, $f(.)$ was the identity function:

$$\begin{aligned} g_x &= f(x) + c_x \\ g_y &= f(y) + c_y \end{aligned} . \tag{8}$$

Normally, the confidence in 6D is calculated in 3D space, but this is very troublesome, so the point was converted to 2D coordinates for calculation. The Euclidean distance was calculated by the difference between the predicted 2D coordinates and the true value. The smaller the distance, the more reliable the prediction. In Eq. (9), d_{th} is the threshold set in advance, such as 30 pixels, and

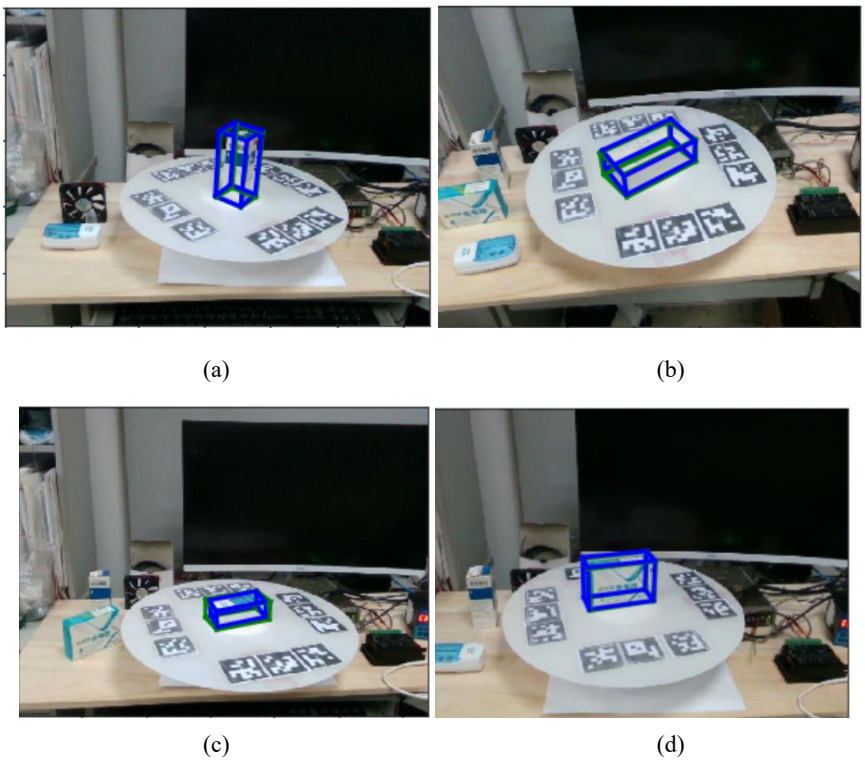
α is the over parameter:

$$c(x)=\begin{cases}(e^{\alpha(1-D_T(x)/d_{th})}-1)/(e^{\alpha}-1), & if\ D_T(x)<d_{th}\\ 0 & otherwise\end{cases}. \tag{9}$$

The loss function of training can be simplified as

$$l_{total}=\lambda_{pt}l_{pt}+\lambda_{conf}l_{conf}+\lambda_{id}l_{id} \tag{10}$$

where l_{pt}, l_{conf}, and l_{id} are the coordinate point, confidence, and classified loss respectively. The preceding coefficients λ are the weights of each loss. The recognition results of four objects in Fig. 8 are shown in Table 1.



(a) (b)

(c) (d)

Figure 8. Grab objects.(a) NAIJIE. (b) LEAP. (c) CHARGE. (d) HYCOSAN.

Table1. Object Pose Recognition Results

	LEAP	HYCOSAN	CHARGE	NAIJIE
5pix	100%	98.05%	98.51%	100%
10% Diameter	37.80%	37.23%	28.29%	39.56%
5cm 5°	70.49%	37.47%	58.31%	76.94%
2D error	2.5640pix	2.7225pix	2.5154pix	2.5661pix
Diameter error	0.0205m	0.0236m	0.0401m	0.0246m
Corner error	4.0901pix	4.0553pix	2.5154pix	3.6938pix
Spatial error	0.2036m	0.0243m	0.0499m	0.0244m
Angle error	4.2519°	4.0553°	6.5344°	4.0933°

Motion planning is a problem in many engineering applications, such as robotics [31-33], navigation, and autonomous driving [34-36]. The essential problem in motion planning is to avoid obstacles and find a path to connect the start and target locations. An improved PRM [18] like algorithm 1 is used in our system:

1:**while** len(vi) < npoints **do**
2: rand(v_i)
3: **if** $v_i \in C_{free}$
4: $V \leftarrow V \cup v_i$
5: $i \leftarrow i+1$
6:**end**
7:**for** all $v_i \in V$ **do**
8: $N \leftarrow$ the closet neighbors of v chosen from V according to disthresh
9: **for** all $v_i \in N$ **do**
10: **if** $((v_i, v_j)) \notin \Pi$ and $LP(v_i, v_j) \neq$ NIL **then**
11: $\Pi \leftarrow \Pi \cup (v_i, v_j)$
12: **end**
13:**end**

Algorithm 1. Probabilistic roadmap planner (PRM)

2.3 Principles of human-robot collaboration

2.3.1 Finite state machine for human-robot collaboration

The switch modes are shown in Fig. 9. The G1 gesture was used to indicate an object. The indication by G1 meant that the robot was required to grasp the object and put it at the target position. G2 was used to switch the robot mode from the indication state to the mapping state. In the mapping state, the hand directly controlled the robot arm's motion. G3 was used to end the mapping state. G4 was used to control the manipulator to open. G1 to G9 means hand was recognized, but when no hand detection, G21 was used to control the manipulator maintain. The user changed the running state of the robot solely by hand. For example, if the current state of the robot was recognition, once the user gave the G2 gesture, the robot switched to mapping, or for G1, the robot switched to the indicative state. Other gestures caused the robot to remain in the recognition state.

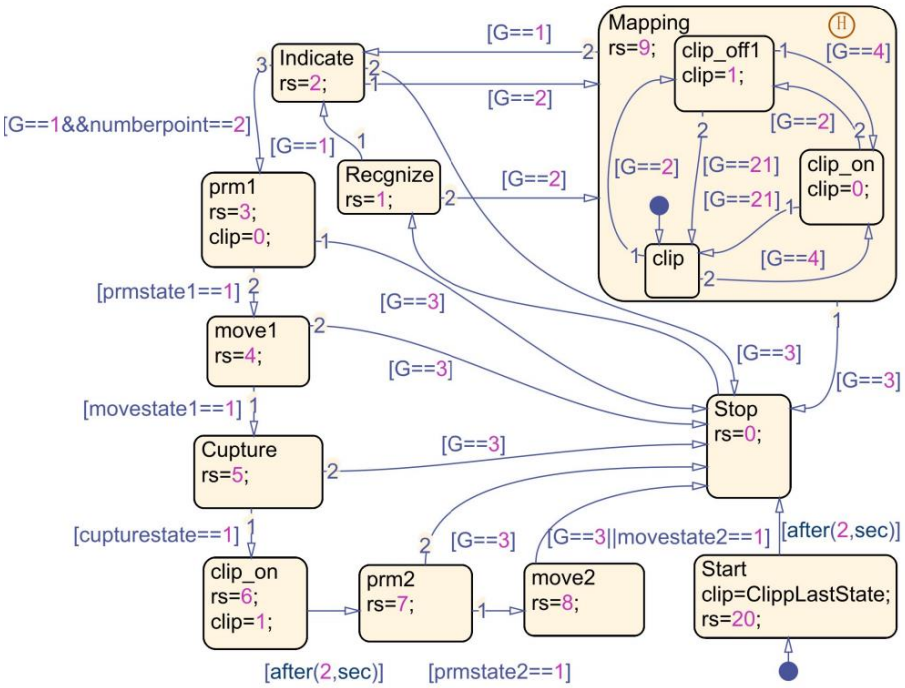


Figure 9. Collaborative interaction framework based on the finite state machine

2.3.2 Human-robot motion mapping

In the operation situation, the posture of the hand had to map onto the end effector. The motion was divided into translation and rotation. In Fig. 12a, the translation part can be expressed by Eq. (11), where ${}^{r}\mathbf{P}_{n}$ is the position of the robot in n times. ${}^{h}\mathbf{U}_{n}$ is the speed. κ is the coefficient

$$. {}^{m}\mathbf{U}_{n} = \kappa({}^{h}\mathbf{P}_{n+1} - {}^{h}\mathbf{P}_{n}) \tag{11}$$

The derivation process can be seen in Eqs. (12) to (17). Where ${}^{h}\mathbf{Q}$ and ${}^{m}\mathbf{Q}$ are the quaternions of the hand and the robot respectively. ${}^{h}\beta$ and ${}^{m}\beta$ are the angles of rotation of the hand and the robot respectively around the rotation axis. In a previous study, electromyography was used to change the flexibility of a pure mapping interaction. For more details, see [37].

Following the quaternion of the hand at the moment n and $n + 1$, the micro rotation difference can be obtained by quaternions:

$$. \Delta^{h}\mathbf{Q}_{n+1} = {}^{h}\mathbf{Q}_{n+1}\,{}^{h}\mathbf{Q}_{n}^{-1} \tag{12}$$

Quaternions can be expressed as

$$. \Delta^{h}\mathbf{Q}_{n+1} = (\cos(\Delta^{h}\beta/2), \mathbf{q}\sin(\Delta^{h}\beta/2)) \tag{13}$$

To solve the rotation angle of the hand:

$$. \Delta^{h}\beta_{n+1} = 2\arccos(\Delta^{h}Q^{0}_{n+1}) \tag{14}$$

The obtained rotation is transferred to the end of the robot arm, and its rotation axis can be expressed as

$$\mathbf{q}=[\Delta^{h}\mathbf{Q}^{1}_{n+1},\Delta^{h}\mathbf{Q}^{1}_{n+2},\Delta^{h}\mathbf{Q}^{1}_{n+3}]^{T}/\sin(\Delta^{h}\beta_{n+1}/2) \tag{15}$$

The rotation angle of the robot can be expressed as

$$\Delta^{m}\beta_{n+1}=\Delta^{h}\beta_{n+1} \tag{16}$$

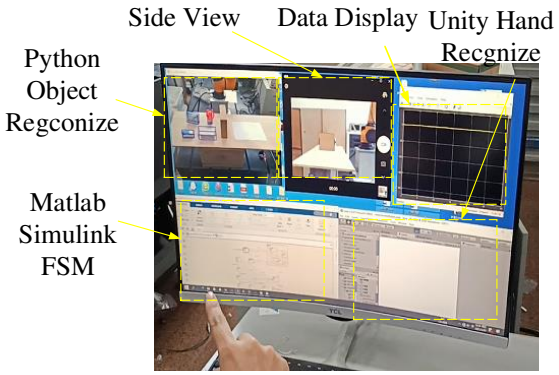
The rotation quaternion of the robot at time n + 1 can be expressed as

$$^{m}\mathbf{Q}_{n+1}=\Delta^{m}\mathbf{Q}_{n+1}\,^{m}\mathbf{Q}_{n} \tag{17}$$

3. Results and discussion

The operation interface of the experiment is shown in Fig. 10a. The gestures were recognized by Leap-Motion. After the neural network processing and geometric computation in Unity, the gesture was passed to Python and Simulink via a universal datagram protocol (UDP). Leap-Motion was connected to a computer via a serial port with a recognition speed of 60 Hz. YOLO 6D running in Python was used to recognize the object posture. The CPU was an Intel i7-4702, the graphics card was a GTX 1060, and the object's recognition frequency was 2 Hz. The robot used AUBO i5 for transmission through a transmission control protocol and Simulink. An OnRobot 6-degrees-of-freedom force sensor, using UDP transmission at a transmission rate of 100 Hz, was used to limit the contact force. To play a protective role, the maximum contact pressure was set to 10 N. All the data were run on a computer, and the calculated trajectories were returned to the AUBO robot.

The steps in the experiment are shown in Fig. 10b. The red dot is the indicating position of the finger. When the object was selected and the target position was determined, it turned into a red circle and flashed three times. Figure 10b1 indicates the position of the object to be grasped. Figure 10b2 indicates the target location. Figure 10b3: Following the robot position and the object's position and obstacle, the trajectory to the object's position started to be planned. Figure 10b4: The robotic arm moved to the objects. Figure 10b5: The mechanical gripper closed and grabbed the object. Figure 10b6: Under the current position, target position, and obstacles of the robot, the trajectory was planned. Figure 10b7: The robot moved the object 10 cm above the target position. Figure 10b8: The operator clenched a fist to start the mapping stage. The position of the end of the hand and the *z*-axis rotation of the vertical ground were mapped to the end of the robot arm to have the robot make a small amount of movement. This stage determined the arrival of objects on the desktop based on the main and side cameras and force sensors. Figure 10b9: With the open hand, the object was set, and the end of the robot arm was raised.



(a)

(b)

Figure 10. Experimental operation procedure

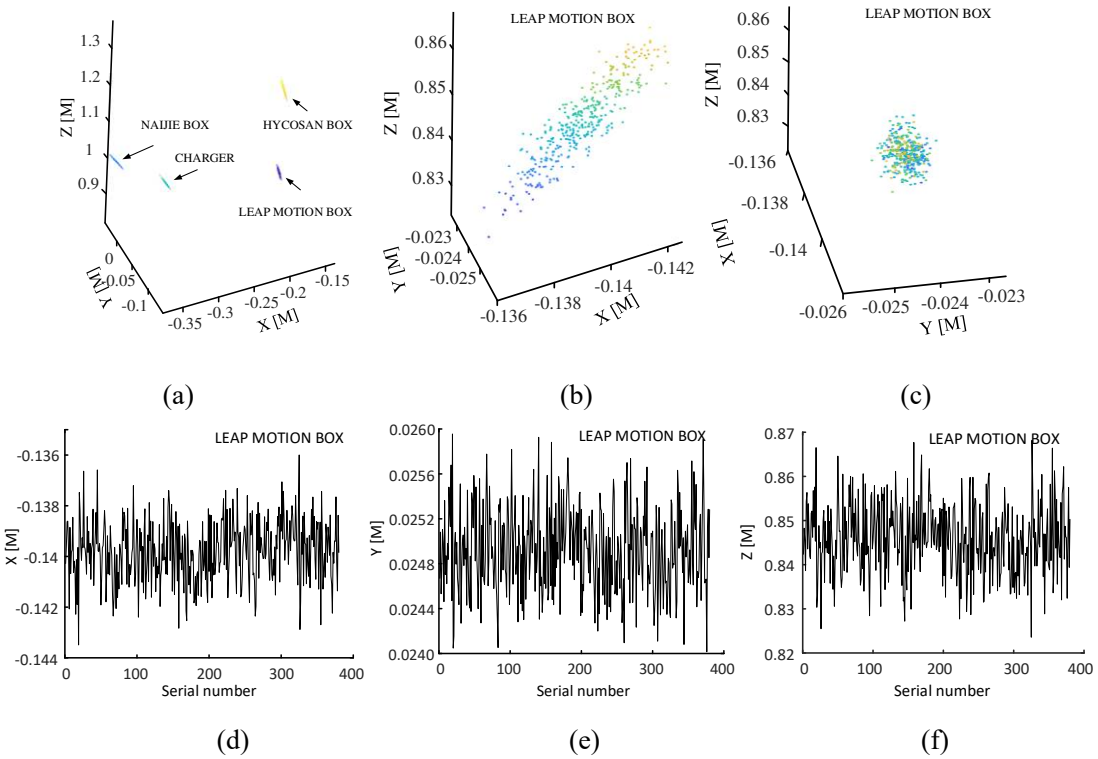


Figure 11. (a) Coordinates of the four objects used in the experiment. (b) and (c) Leap-Motion spatial location. (d), (e), and (f) The X, Y, and Z coordinate positions of the Leap-Motion box respectively, showing the jitter of the object.

The 3D coordinates of the four generated objects are shown in Fig. 11a; 370 frames were collected. The probabilistic position of each object was shaped like an elliptical sphere. Figures 11b and 11c show the 3D spatial position of the Leap-Motion box, which looks like an ellipse from the side and a disk from the camera's perspective. Figures 11d, 11e, and 11f show that the jitter of the Leap-Motion box had a 1.23-mm standard deviation; the maximum and minimum difference in the *x*-axis was 7.49 mm. In the *y* axis, there was a 0.38-mm standard deviation and a 1.94-mm maximum and minimum difference. In the *z*-axis, there was an 8.18-mm standard deviation and a 44.61-mm maximum and minimum difference. The jitter could have been caused by camera shake, changes in ambient light, or another cause. An object attitude estimation by YOLO 6D is based on 2D images, so such estimations always have a certain deviation in 3D space. However, the position in 3D space was converted into pixel coordinates.

The world coordinate went to the camera coordinate, then to the image coordinate, and finally to the pixel coordinate, where *Xw*, *Yw*, and *Zw* were the world coordinates of an object. In Eq. (18), *R* is the rotation matrix and *T* is the translation matrix, both constituting the camera external parameters; f is the focal length; dx and dy are the number of units of length of each pixel in the *x* and *y* directions; u_0 and v_0 are the number of horizontal and vertical pixels that differ

between the pixel coordinates of the center of the image, and γ is the distortion factor:

$$Z_C \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{dx} & \gamma & u_0 \\ 0 & \frac{1}{dy} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} [R|T] \begin{bmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{bmatrix} . \tag{18}$$

The pixel results obtained are shown in Fig. 12. Figure 12a shows that all objects were located at small points. Figures 12b and 12c show that the variance in the X direction observed numerically was 0.45 pixels, and the maximum and minimum pixel difference was 2.45 pixels. The variance in the Y direction was 0.19 pixels, and the maximum and minimum pixel difference was 1.18 pixels. It can be seen that the accuracy of the plane-based object attitude recognition was acceptable after it was converted into pixels.

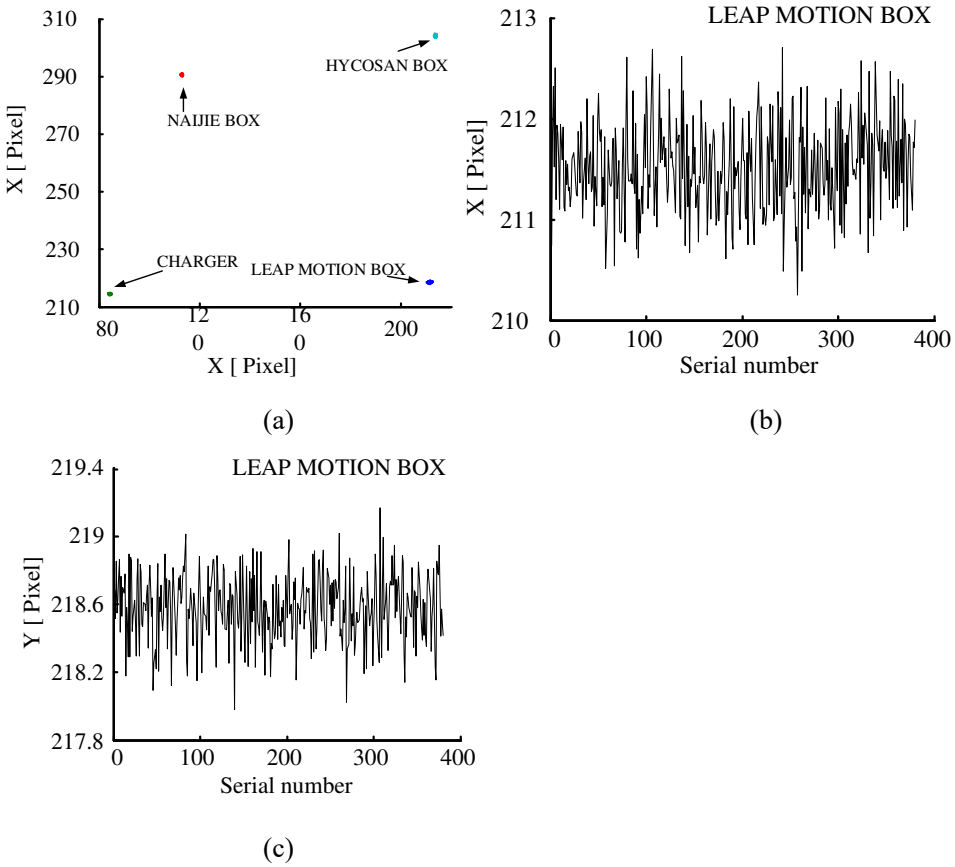


Figure 12. (a) Pixel coordinates of the four objects used in the experiment. (b), (c) X, Y pixel coordinate positions of the Leap-Motion box.

$$\mathbf{P}^n = F(\mathbf{P}^o, \Delta X, \Delta Y, H, W, R) \tag{19}$$

$$\mathbf{P}_y^n \sim \mathbf{P}_y^o + \alpha H + \beta W \tag{20}$$

$$\mathbf{P}_x^n \sim \mathbf{P}_x^o + \xi \Delta X^2 (\Delta Y + \alpha H + \beta W). \tag{21}$$

The four points approximately correspond to the actual position In Figs. 13a and 13b. However, the point positions were somewhat different from the actual positions, so the spatial object mapped to the colour image had to be transformed. A large and flexible mechanical grip on a robot arm has less of a requirement for an accurate position, but the mechanical grip in this study was small and rigid, so the position had to be corrected. Although the position was obtained by the junction point cloud, the geometric size of the object had to be further considered to be corrected.

To simplify the process, a correction equation was devised. (Based on the actual situation in this study, the positions of several of the objects did not differ greatly). The position of the object could be obtained by combining pixel coordinates, depth data, rotation, and object size. As shown in Eqs. (19) to (21), the corrected displacement of the object $\mathbf{P}^n$ was related to the measured displacement $\mathbf{P}^o$; the displacement of the object from the pixel centers $\Delta X, \Delta Y$; the width and height of the object H and W; and the rotation R of the object. The offset in the Y direction was corrected by Eq. (20), which was mainly related to the height and width of the object. The offset in the Z direction was corrected by Eq. (21), which is nonlinear, so it used ΔX^2. The scale factor was ξ. The results modified are shown in Fig. 13c and 13d, which also show that the object position was close to the actual position and met the grasping requirement.

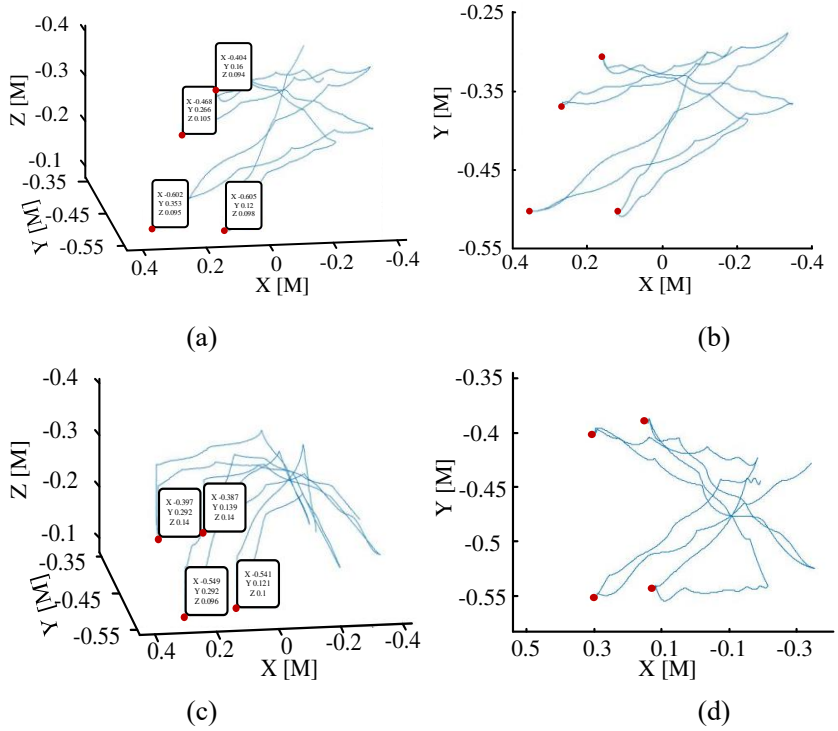


Figure 13. Automatic motion trajectory of the AUBO robot

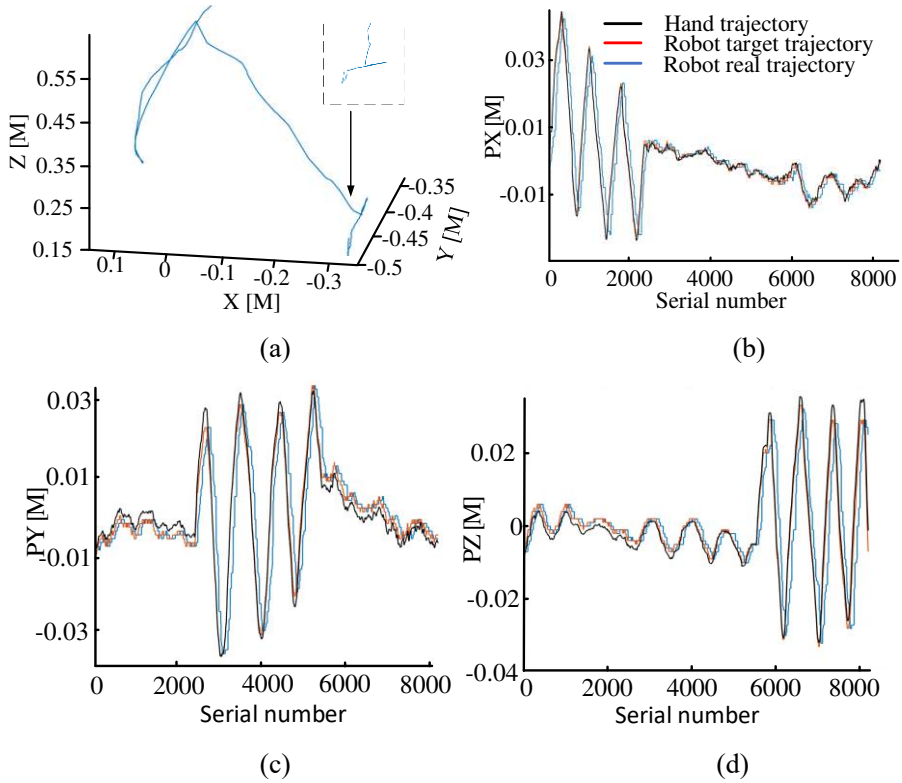


Figure 14. Motion trajectory of the AUBO manipulator

The complete trajectory is shown in Fig. 14a. The robot avoided the obstacle by itself. Only the placement (inside the dotted box) was completed by gestures directly mapped to the user. The manual mapping part is the final stage of fine-tuning. To observe the situation of the robot following the hand, the reciprocating movement is added. As shown in Fig. 14b,14c and 14d the hand trajectory is relatively round. The displacement target of the robot is closed to the hand trajectory. Although the delay of the robot causes some sense of contour in the actual running trajectory, the robot can follow the local motion well on the whole. In comparison, whole-process location mapping [38] requires a user to operate all trajectories, including movement, grasping, obstacle avoidance, and placement, which is a heavy burden on the user. In some situations, automatic operation can be achieved, such as peg-in-hole assembly [39] with complex requirements. But some scenes require a user to participate. For example, take a waiter in a restaurant: There are many studies on motion obstacle avoidance for restaurant food delivery robots. The food can be delivered to customers, but cannot be placed [40]. In a Chinese restaurant, there are usually a few more dishes than in other restaurants, which can cause an overlap problem. Also, it may be necessary to consider who is has the highest status at the table, which may influence the dish placement position; that undoubtedly increases the difficulty of placement. HRC grasping and placing, seemingly simple, is a research direction full of challenges and rich rewards.

4. Conclusion

In this study, Leap-Motion was used to collect hand data, and then nine gestures were taught to and recognized by the neural network. The control mode combined the gestures and an FSM. Based on YOLO 6D, the position of an object can be determined and corrected by combining depth data with geometric transformation.Then, by combining the depth data and an improved PRM, the robot trajectory was generated to achieve obstacle avoidance. Results show that in the interactive mode, most of the tracking can be completed by a robot, and only a small part requires user participation, which reduces the burden on the user. Future work will focus on the user experience and systems of force feedback. Also, intelligent learning capabilities may be added that automatically learn the habits of the user placement. Finally, a mobile module could be integrated to increase the working range of the robot. The present study will further promote robot intelligence and enhance the degree of HRC.

Fundings

The project is supported by The National Key Research and Development Program of China （No.2017YFB1301400), Industrial verification platform and performance evaluation of precision machine tool spindle bearing of The National Key Research and Development Program of China （No. 2018YFB2000500) and Graduate Scientific Research and Innovation of Chongqing (NO. CYB19062).

References

1. Tsitsimpelis I, Taylor CJ, Lennox B et al (2019) A review of ground-based robotic systems for the characterization of nuclear environments. Progress in Nuclear Energy 111: 109-124. https://doi.org/
2. Birk A, Doernbach T, Mueller C et al (2018) Dexterous underwater manipulation from onshore locations: Streamlining efficiencies for remotely operated underwater vehicles. IEEE Robotics Automation Magazine 25: 24-33. https://doi.org/
3. Chen Z, Huang F, Yang C et al (2020) Adaptive Fuzzy Backstepping Control for Stable Nonlinear Bilateral Teleoperation Manipulators With Enhanced Transparency Performance. Ieee T Ind Electron 67: 746-756. https://doi.org/10.1109/tie.2019.2898587
4. Bachmann ER, McGhee RB, Yun X et al (2001) Inertial and magnetic posture tracking for inserting humans into networked virtual environments. Proceedings of the ACM symposium on Virtual reality software and technology: 9-16. https://doi.org/
5. Filippeschi A, Schmitz N, Miezal M et al (2017) Survey of motion tracking methods based on inertial sensors: A focus on upper limb human motion. Sensors 17: 1257. https://doi.org/
6. Qi J, Jiang G, Li G et al (2020) Surface EMG hand gesture recognition system based on PCA and GRNN. Neural Computing Applications 32: 6343-6351. https://doi.org/
7. Kebria PM, Abdi H, Dalvand MM et al (2018) Control methods for internet-based teleoperation systems: A review. IEEE Transactions on Human-Machine Systems 49: 32-46. https://doi.org/
8. Chakraborty BK, Sarma D, Bhuyan MK et al (2017) Review of constraints on vision-based gesture recognition for human–computer interaction. IET Computer Vision 12: 3-15. https://doi.org/
9. Zeng H, Shen Y, Hu X et al (2020) Semi-autonomous robotic arm reaching with hybrid gaze–brain machine interface. Frontiers in neurorobotics 13: 111. https://doi.org/
10. Lan T-S, Her M-G, Hsu K-S (2003) Virtual reality application for direct-drive robot with force feedback. The International Journal of Advanced Manufacturing Technology 21: 66-71. https://doi.org/
11. Her M-G, Hsu K-S, Yu W-S (2002) Analysis and design of a haptic control system: virtual reality approach. The International Journal of Advanced Manufacturing Technology 19: 743-751.

https://doi.org/
12. Goodrich MA, Crandall JW, Barakova E (2013) Teleoperation and beyond for assistive humanoid robots. Reviews of Human factors ergonomics 9: 175-226. https://doi.org/
13. Liang JS, Chao K-M, Ivey P (2013) VR-based wheeled mobile robot in application of remote real-time assembly. The International Journal of Advanced Manufacturing Technology 64: 1765-1779. https://doi.org/
14. Hayati S, Venkataraman S (1989) Design and implementation of a robot control system with traded and shared control capability. 1989 IEEE International Conference on Robotics and Automation: 1310, 1311, 1312, 1313, 1314, 1315-1310, 1311, 1312, 1313, 1314, 1315. https://doi.org/
15. Bauer AS, Schmaus P, Albu-Schäffer A et al (2018) Inferring semantic state transitions during telerobotic manipulation. 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS): 1-9. https://doi.org/
16. Du G, Yao G, Li C et al (2019) Natural human–robot interface using adaptive tracking system with the unscented Kalman filter. IEEE Transactions on Human-Machine Systems 50: 42-54. https://doi.org/
17. Yang C, Wang X, Cheng L et al (2016) Neural-learning-based telerobot control with guaranteed performance. IEEE transactions on cybernetics 47: 3148-3159. https://doi.org/
18. Kavraki LE, Svestka P, Latombe J et al (1996) Probabilistic roadmaps for path planning in high-dimensional configuration spaces. international conference on robotics and automation 12: 566-580. https://doi.org/
19. Xue Y, Ju Z, Xiang K et al (2018) Multimodal human hand motion sensing and analysis—A review. IEEE Transactions on Cognitive Developmental Systems 11: 162-175. https://doi.org/
20. Hernandez C, Bharatheesha M, Ko W et al (2016) Team delft's robot winner of the amazon picking challenge 2016. Robot World Cup: 613-624. https://doi.org/
21. Sahin C, Garcia-Hernando G, Sock J et al (2020) A review on object pose recovery: from 3d bounding box detectors to full 6d pose estimators. Image Vision Computing 96: 103898. https://doi.org/
22. Redmon J, Divvala S, Girshick R et al (2016) You only look once: Unified, real-time object detection. Proceedings of the IEEE conference on computer vision and pattern recognition: 779-788. https://doi.org/
23. Tekin B, Sinha SN, Fua P (2018) Real-time seamless single shot 6d object pose prediction. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition: 292-301. https://doi.org/
24. Du GL, Zhang P, Liu X (2016) Markerless Human-Manipulator Interface Using Leap Motion With Interval Kalman Filter and Improved Particle Filter. Ieee Transactions on Industrial Informatics 12: 694-704. https://doi.org/10.1109/tii.2016.2526674
25. Luzanin O, Plancak M (2014) Hand gesture recognition using low-budget data glove and cluster-trained probabilistic neural network. Assembly Automation 34: 94-105. https://doi.org/
26. Wu B-X, Yang C-G, Zhong J-P (2021) Research on Transfer Learning of Vision-based Gesture Recognition. International Journal of Automation Computing: 1-10. https://doi.org/
27. Hodan T, Haluza P, Obdržálek Š et al (2017) T-LESS: An RGB-D dataset for 6D pose estimation of texture-less objects. 2017 IEEE Winter Conference on Applications of Computer Vision (WACV): 880-888. https://doi.org/
28. Engelcke M, Rao D, Wang DZ et al (2017) Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks. 2017 IEEE International Conference on Robotics and Automation (ICRA): 1355-1361. https://doi.org/
29. Fang B, Sun F, Liu H et al (2020) Wearable Technology for Robotic Manipulation and Learning Springer
30. Timmi A, Coates G, Fortin K et al (2018) Accuracy of a novel marker tracking approach based on the low-cost Microsoft Kinect v2 sensor. Medical Engineering & Physics 59: 63-69. https://doi.org/10.1016/j.medengphy.2018.04.020
31. Zhang T, Su J (2018) Collision-free planning algorithm of motion path for the robot belt grinding system. Int J Adv Robot Syst 15: 1729881418793778. https://doi.org/
32. Ye G, Alterovitz R (2017) Demonstration-Guided Motion Planning. international symposium on robotics: 291-307. https://doi.org/
33. Ichter B, Harrison J, Pavone M (2018) Learning Sampling Distributions for Robot Motion Planning. international conference on robotics and automation: 7087-7094. https://doi.org/
34. Mu B, Giamou M, Paull L et al (2016) Information-based active SLAM via topological feature graphs. Conference on Decision and Control: 5583-5590. https://doi.org/
35. Wang P, Gao S, Li L et al (2019) Obstacle Avoidance Path Planning Design for Autonomous Driving

Vehicles Based on an Improved Artificial Potential Field Algorithm. Energies 12: 2342. https://doi.org/
36. Francis A, Faust A, Chiang H-TL et al (2019) Long-Range Indoor Navigation with PRM-RL. ArXiv 1902.09458
37. Zhao X, Chen X, He Y et al (2019) Varying Speed Rate Controller for Human–Robot Teleoperation Based on Muscle Electrical Signals. IEEE Access 7: 143563-143572. https://doi.org/10.1109/ACCESS.2019.2944794
38. Handa A, Van Wyk K, Yang W et al (2020) DexPilot: Vision-Based Teleoperation of Dexterous Robotic Hand-Arm System. 2020 IEEE International Conference on Robotics and Automation (ICRA): 9164-9170. https://doi.org/
39. Song J, Chen Q, Li Z (2021) A peg-in-hole robot assembly system based on Gauss mixture model. Robotics Computer-Integrated Manufacturing 67: 101996. https://doi.org/
40. Kong L, Peng X, Chen Y et al (2020) Multi-sensor measurement and data fusion technology for manufacturing process monitoring: a literature review. International Journal of Extreme Manufacturing 2: 022001. https://doi.org/

Figures

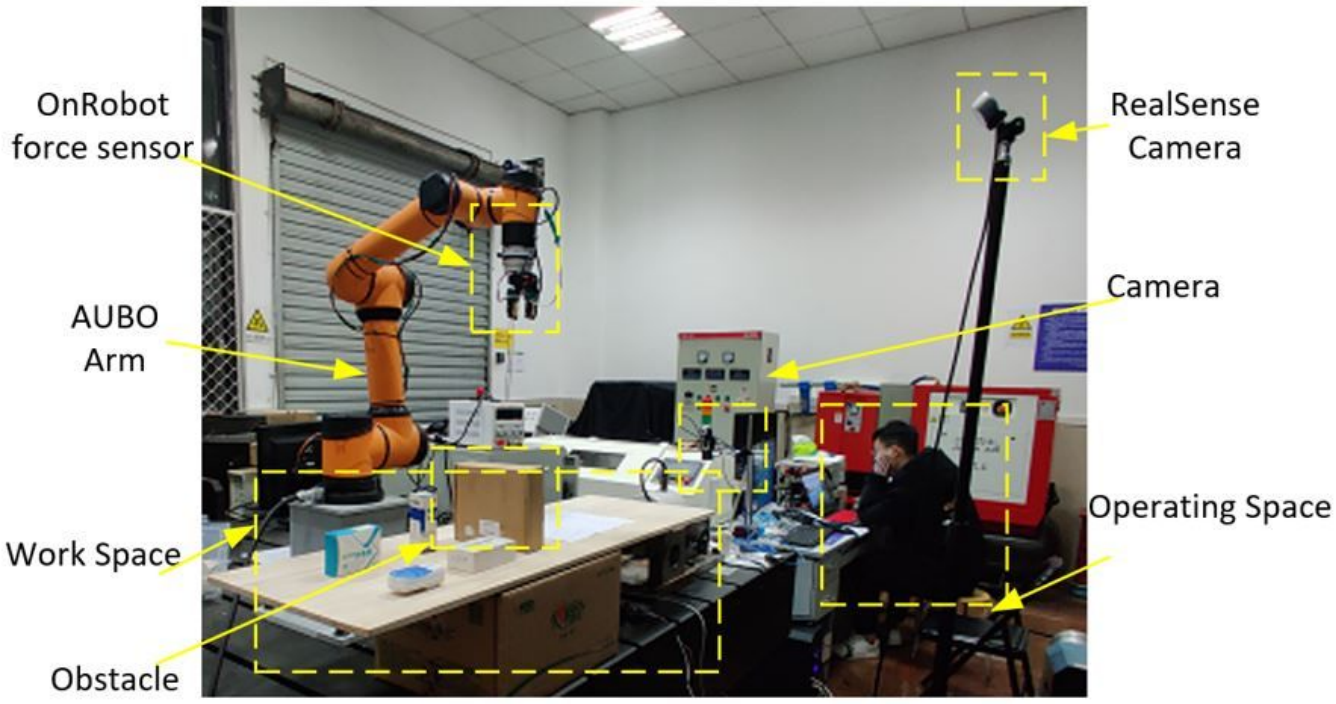


Figure 1

Scene of human-robot interaction

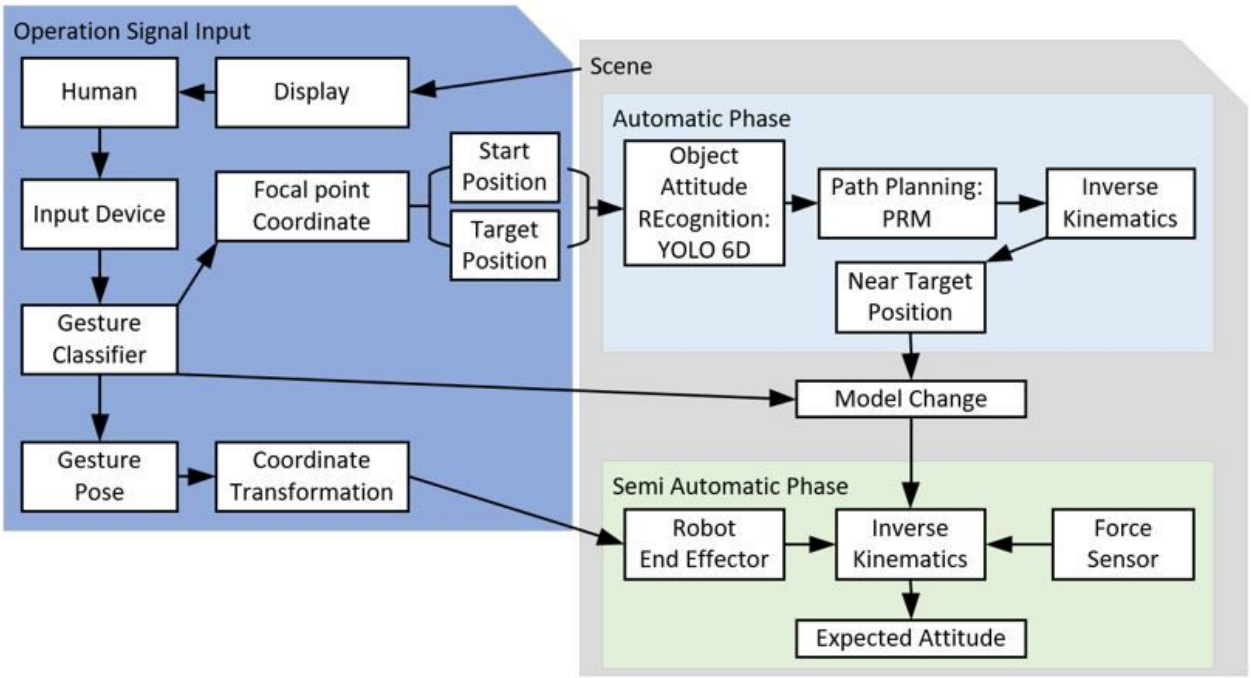
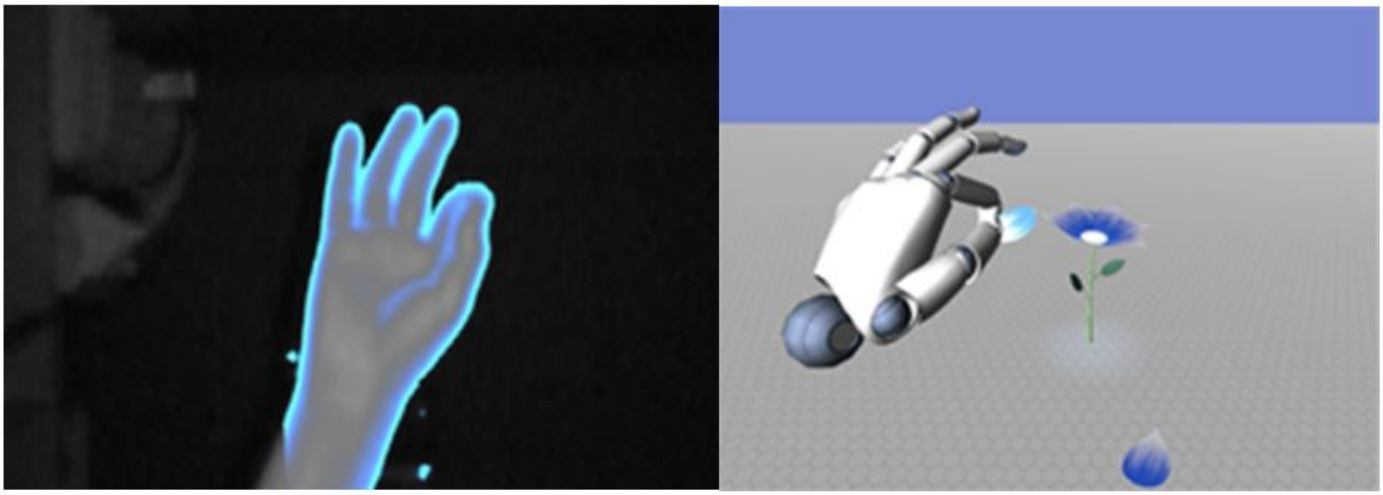


Figure 2

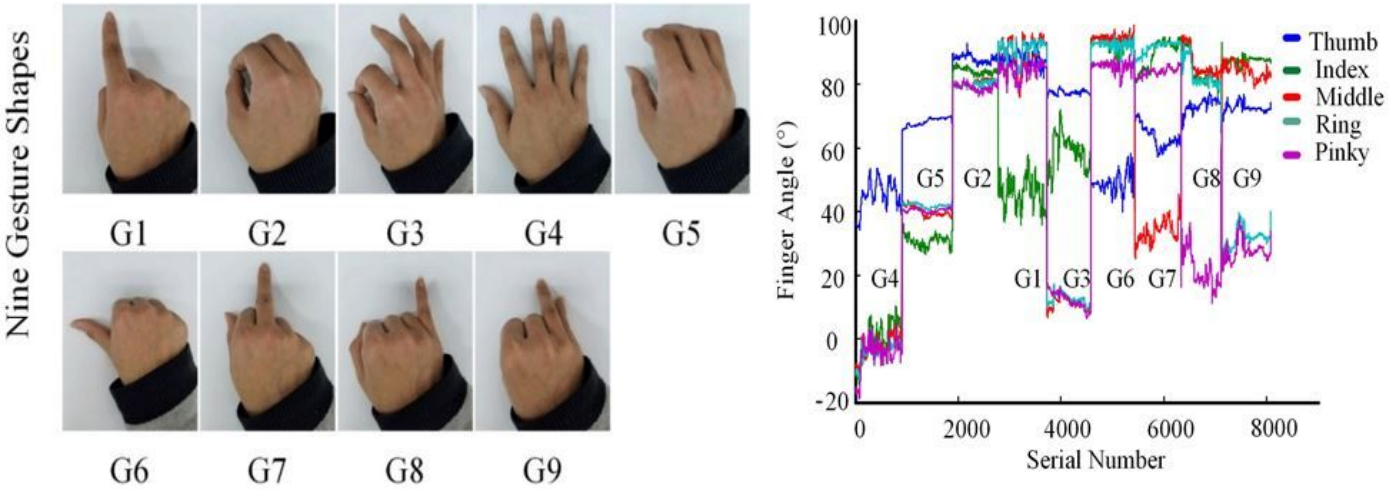
Interaction system structure



(a) (b)

Figure 3

Immersive interaction with (a) Leap-Motion and (b) Unity



(a) (b)

Figure 4

(a) Gesture classifications. (b) G1 to G9 correspond to the bending angles of the fingers.

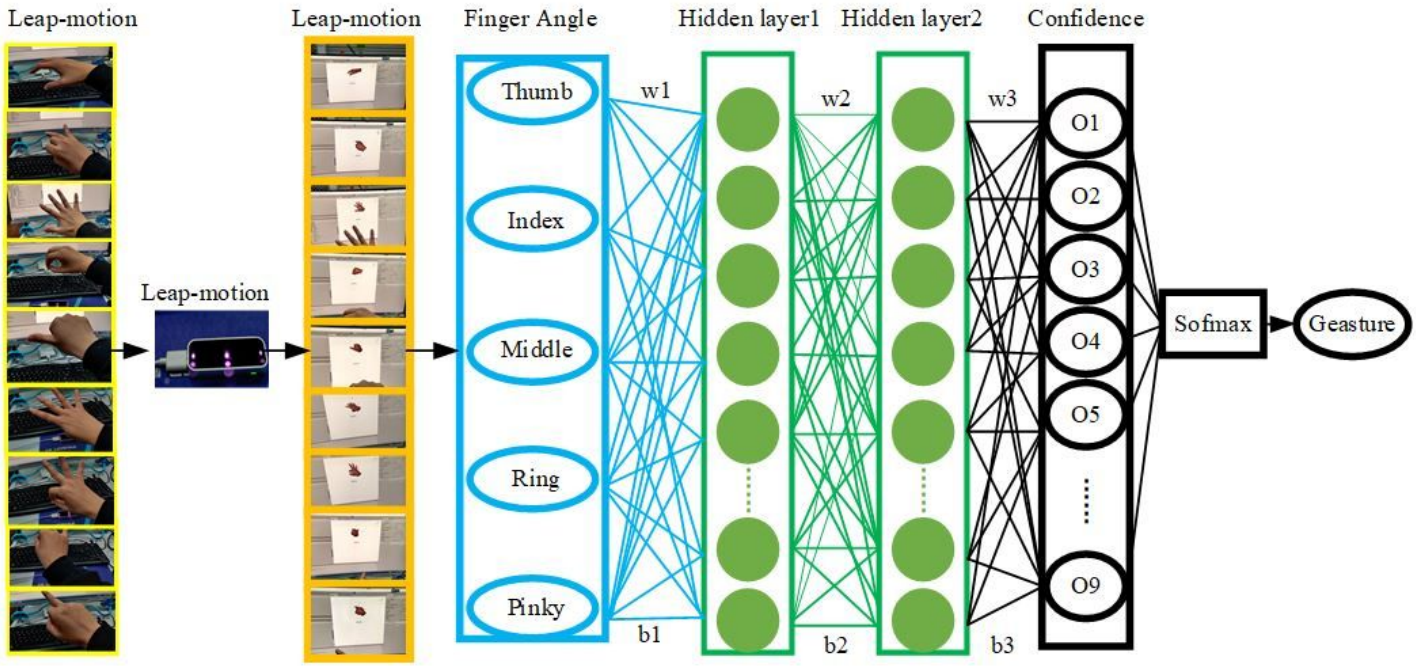


Figure 5

Structure of the finger recognition neural network

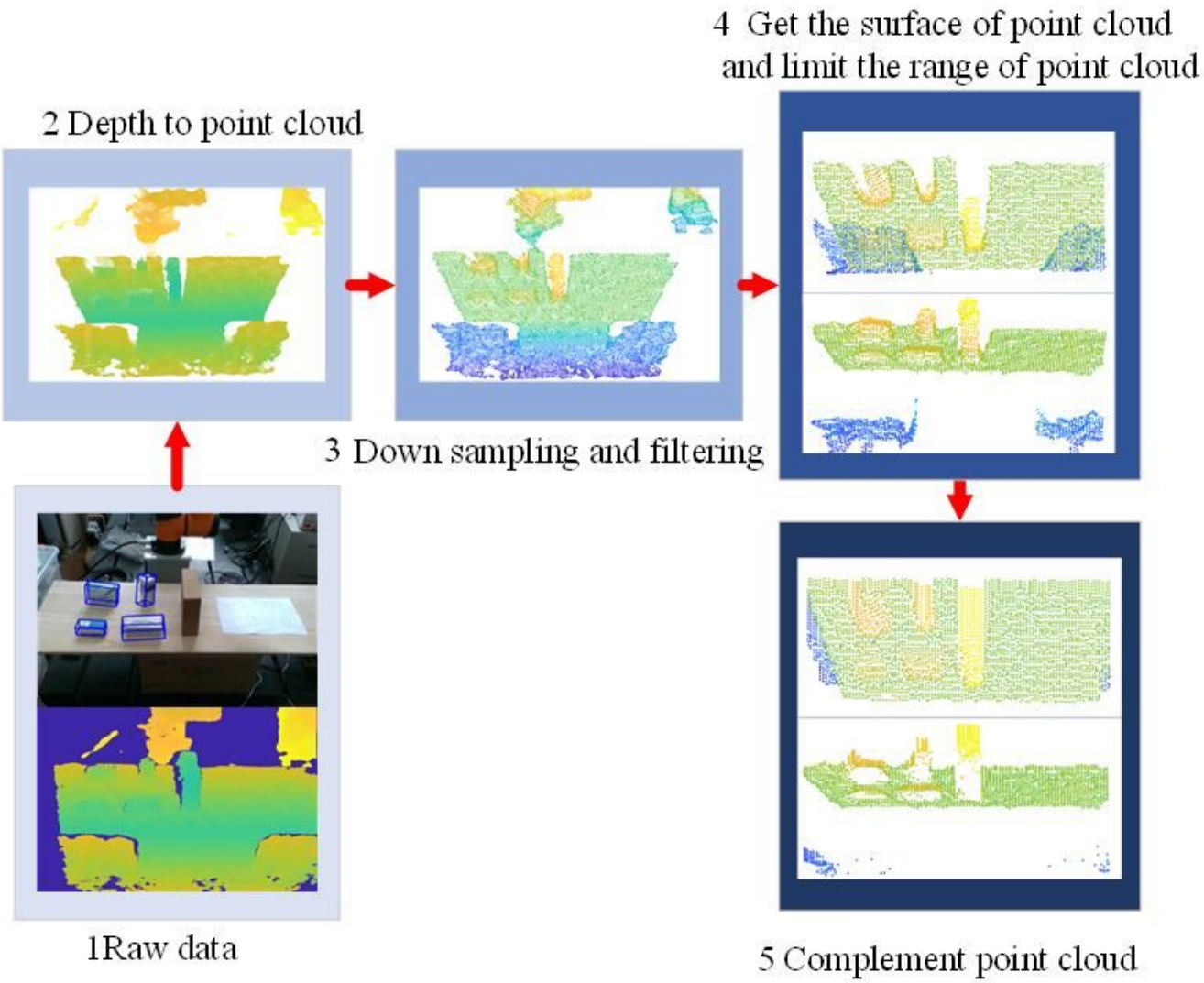


Figure 6

Scene generation using a depth camera

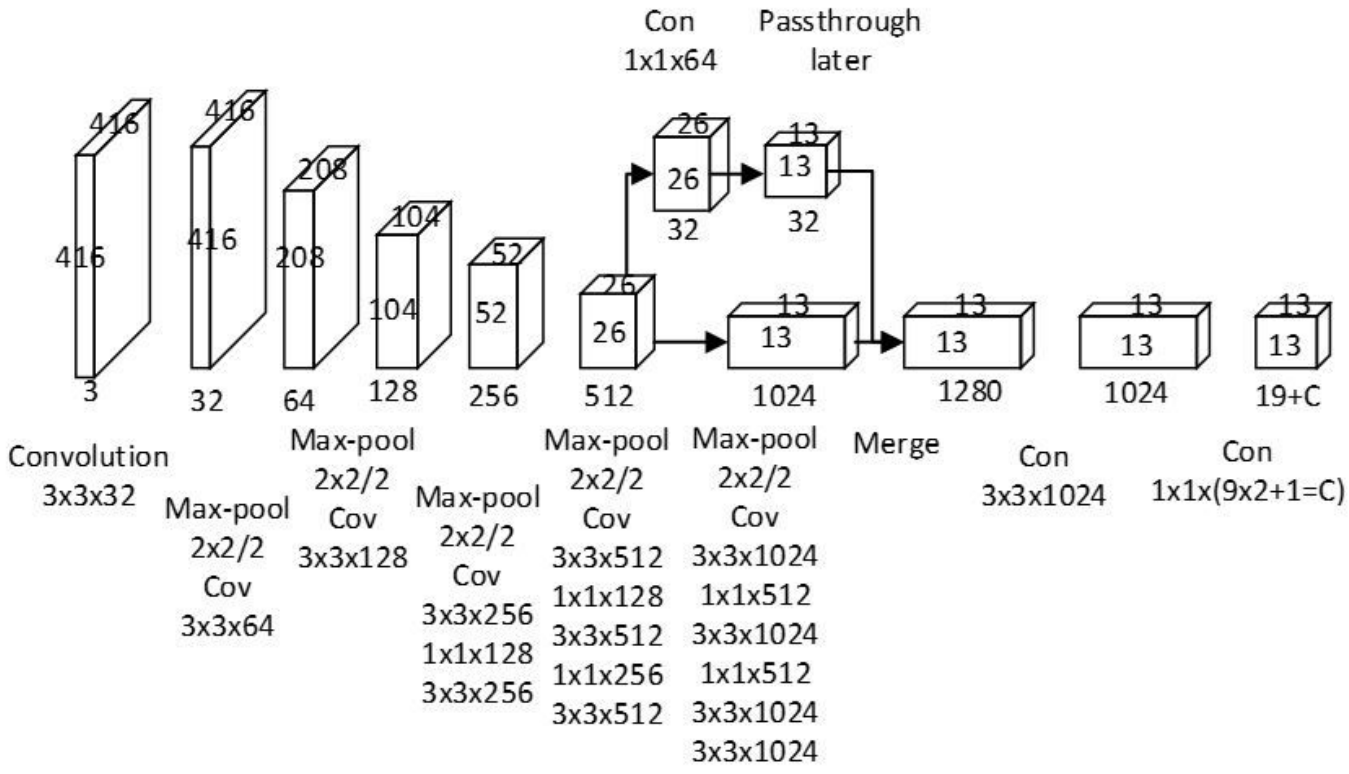


Figure 7

YOLO 6D network structure

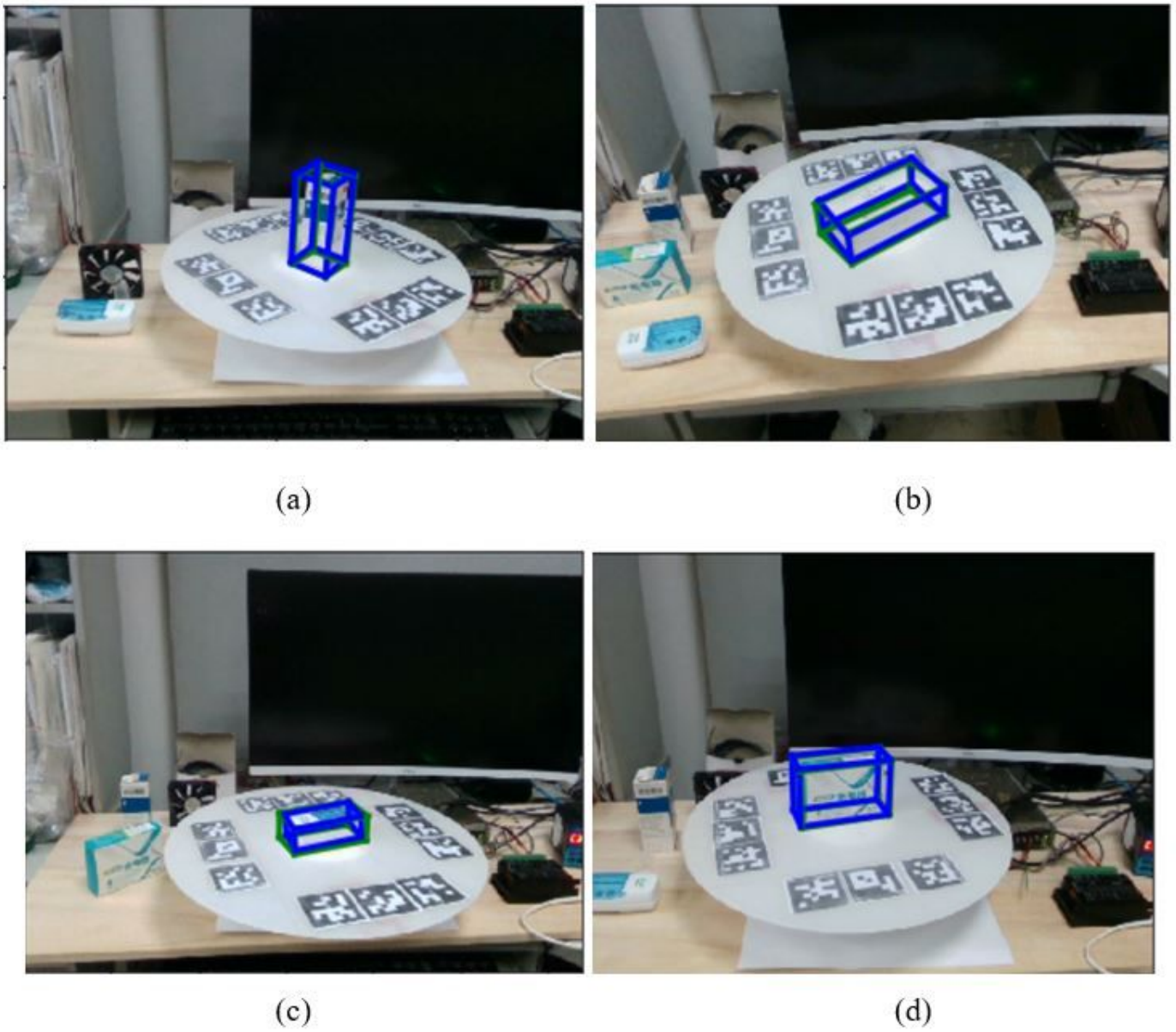


Figure 8

Grab objects.(a) NAIJIE. (b) LEAP. (c) CHARGE. (d) HYCOSAN.

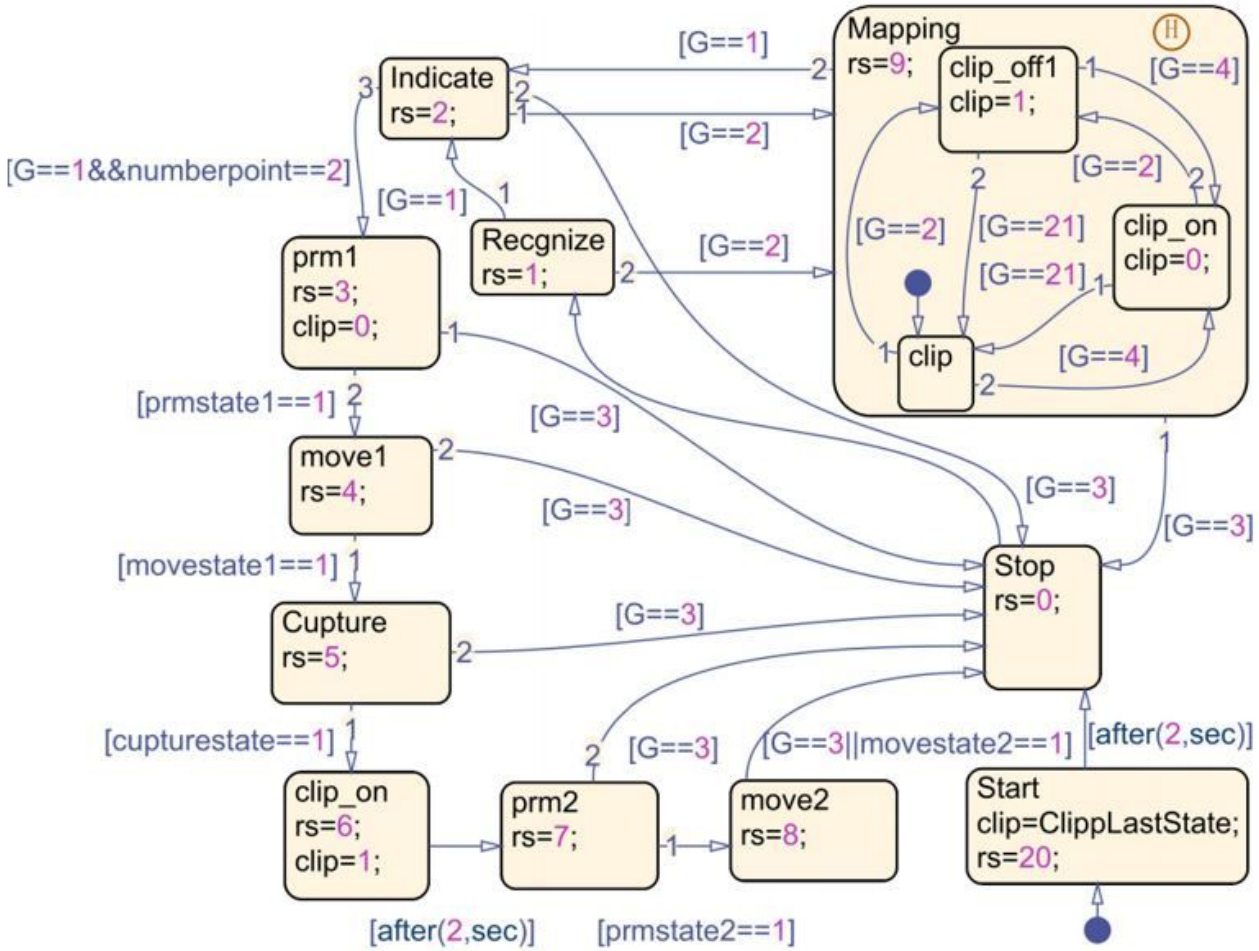


Figure 9

Collaborative interaction framework based on the finite state machine

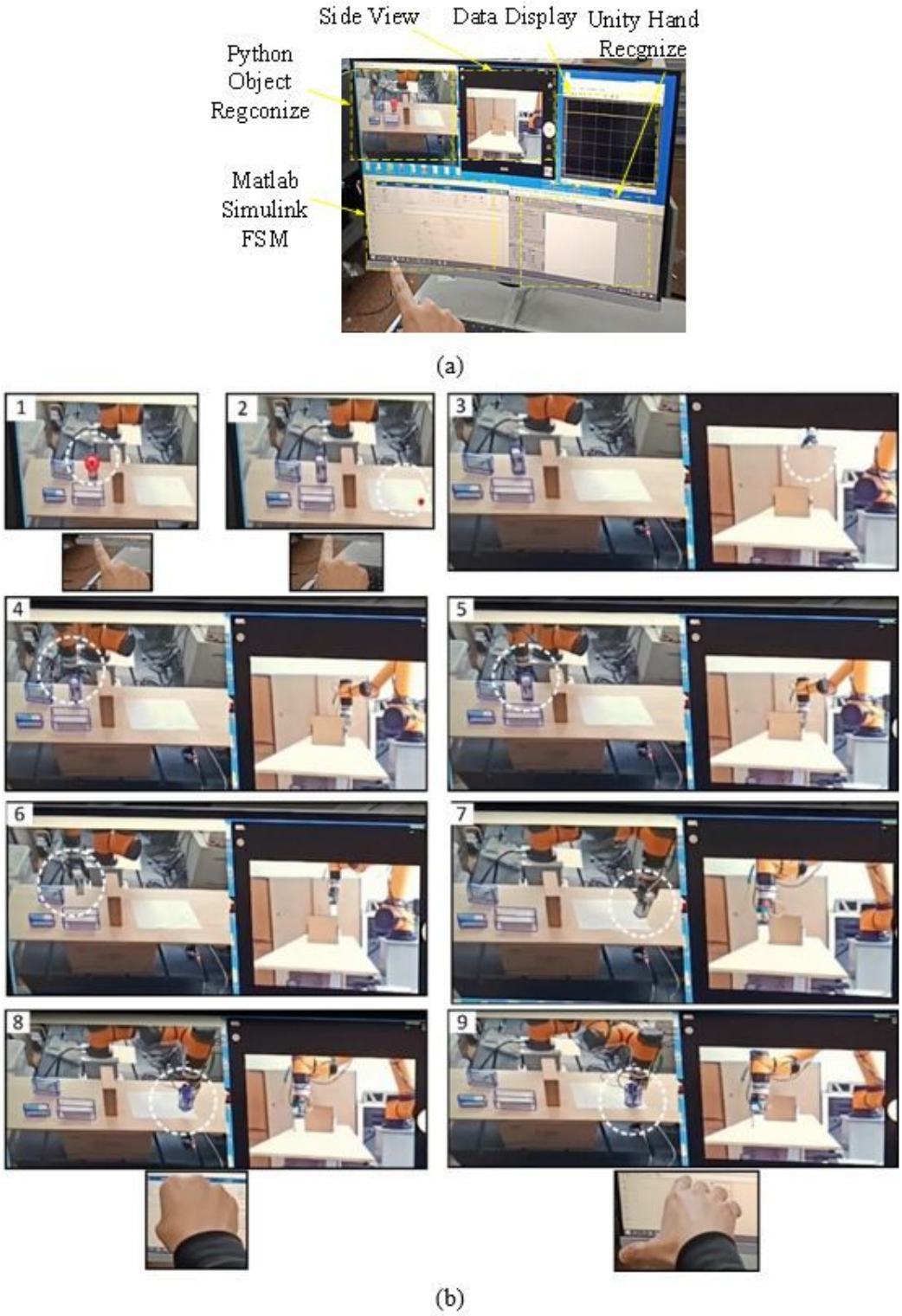


Figure 10

Experimental operation procedure

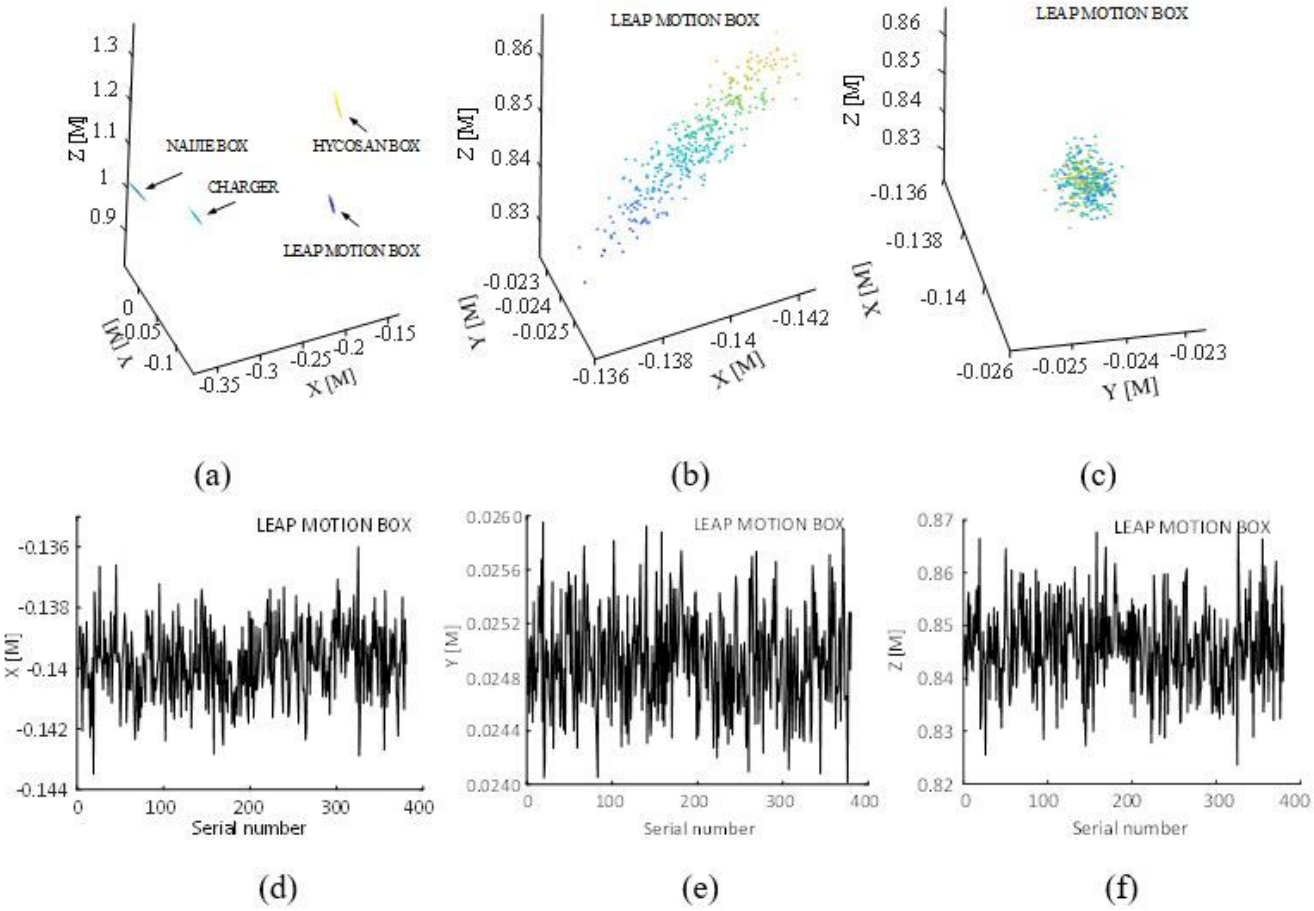


Figure 11

(a) Coordinates of the four objects used in the experiment. (b) and (c) Leap-Motion spatial location. (d), (e), and (f) The X, Y, and Z coordinate positions of the Leap-Motion box respectively, showing the jitter of the object.

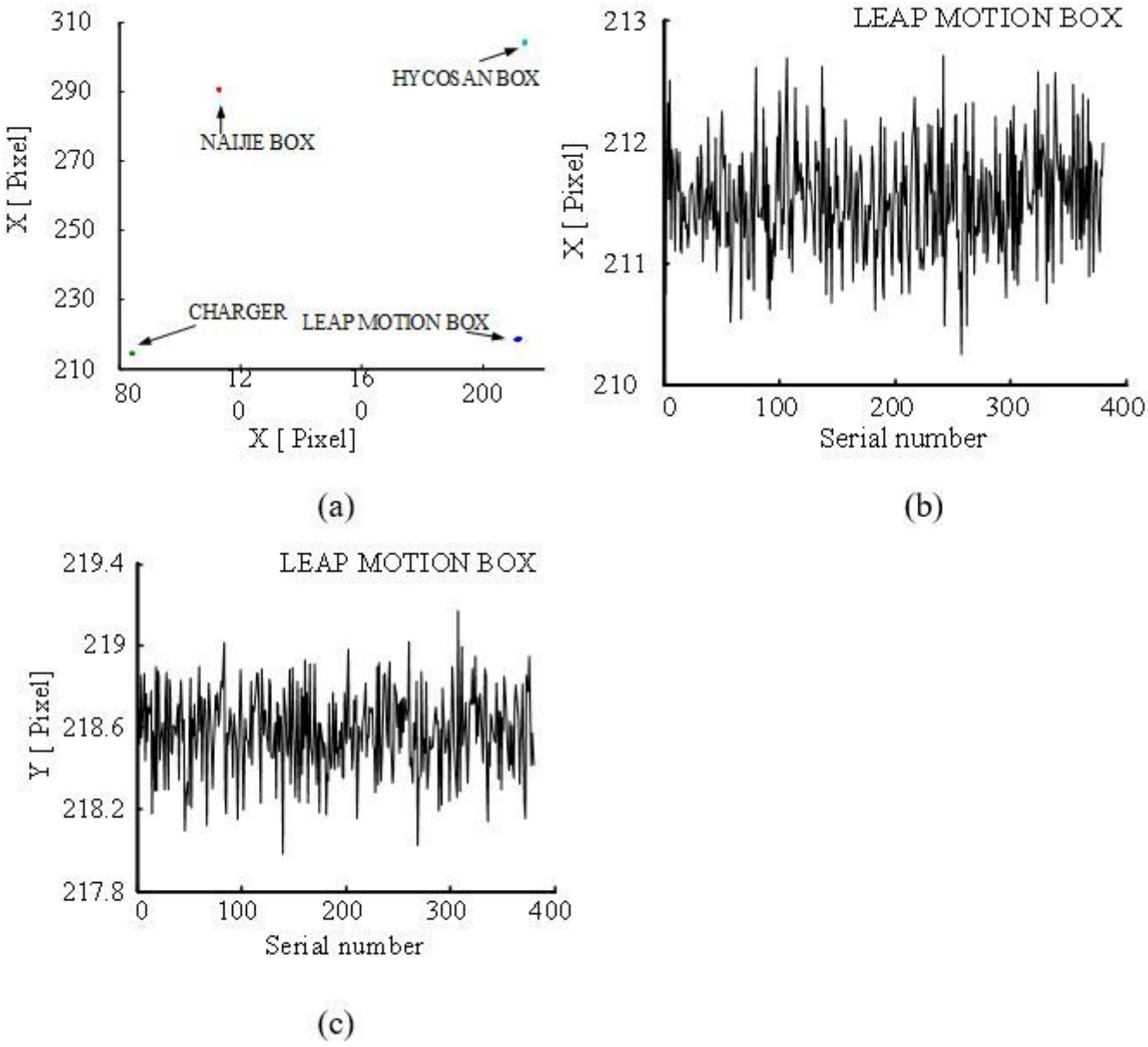


Figure 12

(a) Pixel coordinates of the four objects used in the experiment. (b), (c) X, Y pixel coordinate positions of the Leap-Motion box.

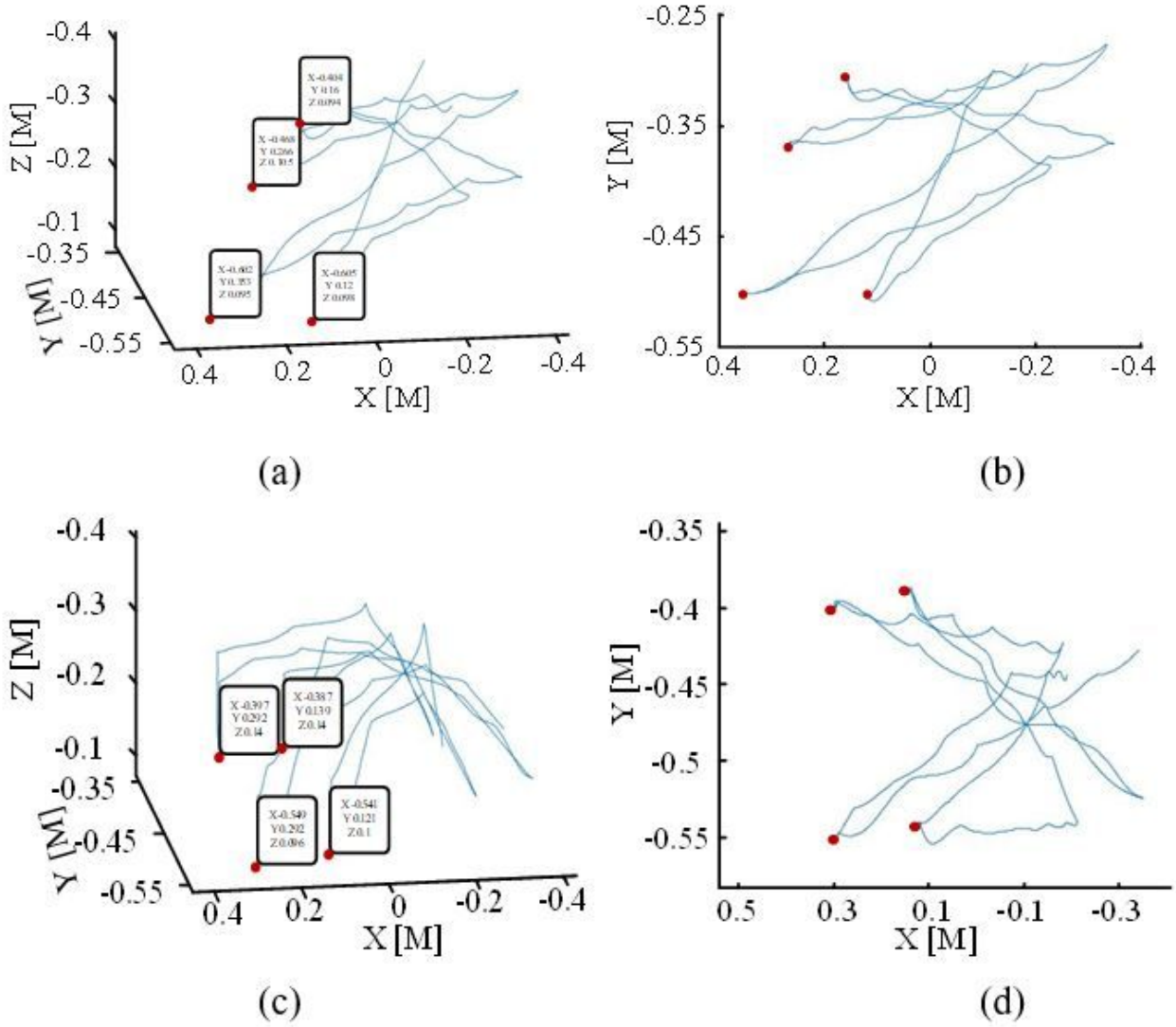


Figure 13

Automatic motion trajectory of the AUBO robot

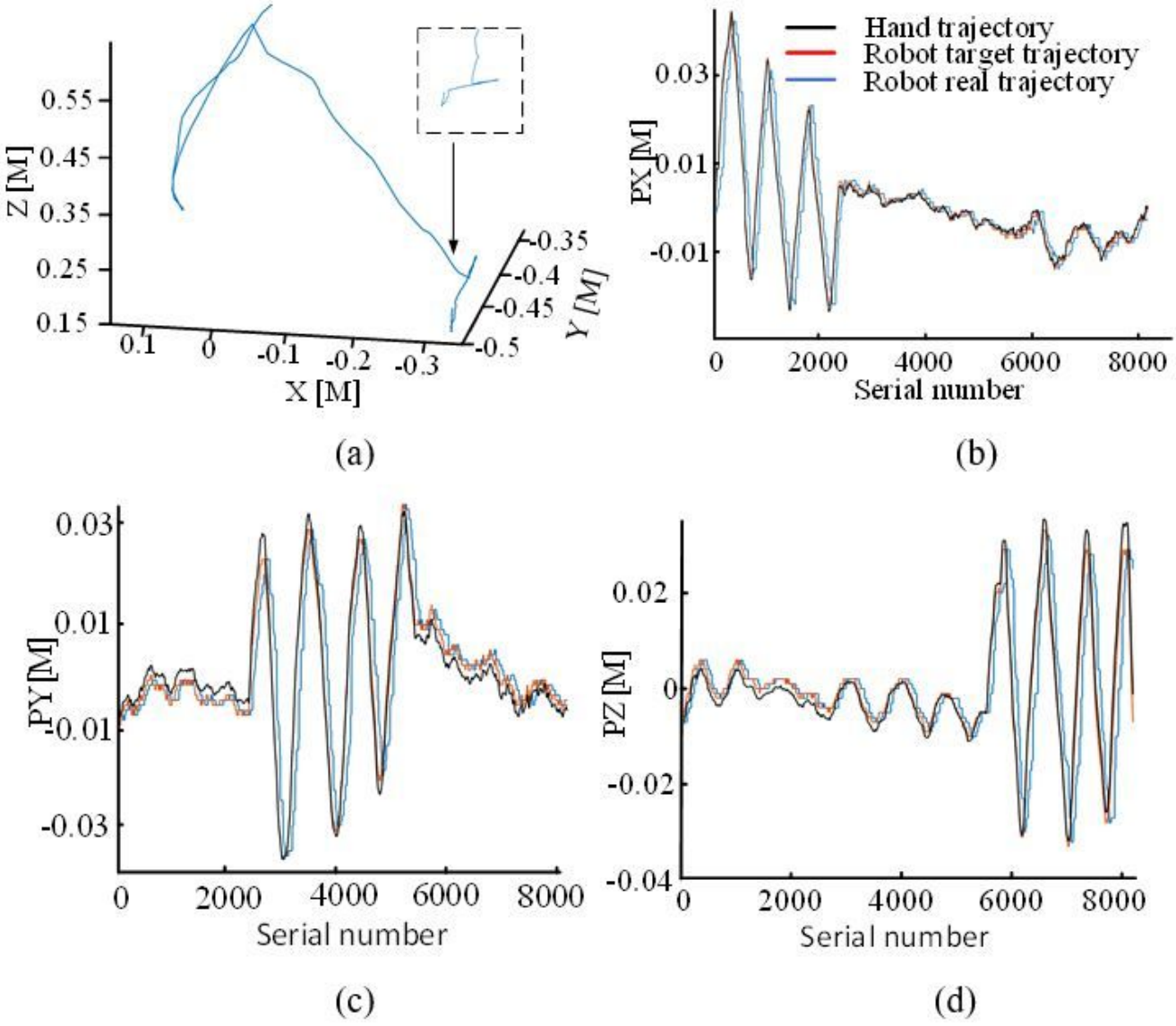


Figure 14

Motion trajectory of the AUBO manipulator