







Article

# Color changing object recognition and grabbing technology based on crystal butterfly algorithm and adaptive imitation

Zuoxun Wang<sup>1,2</sup>  , Chuanyu Cui<sup>1,2,3</sup>  , Jinxue Sui<sup>1,2</sup>, Changkun Guo<sup>1,2</sup>


Show more 

 Outline |  Share  Cite

<https://doi.org/10.1016/j.isci.2024.110457> 

[Get rights and content](#) 

Under a Creative Commons [license](#) 

 open access

## Highlights

- Crystal Butterfly tracks dynamic color-changing nodes, enhancing data richness
- CDR in Crystal Butterfly improves dynamic color recognition via path memory
- AISP enhances grasping adaptability in complex environments

PDF

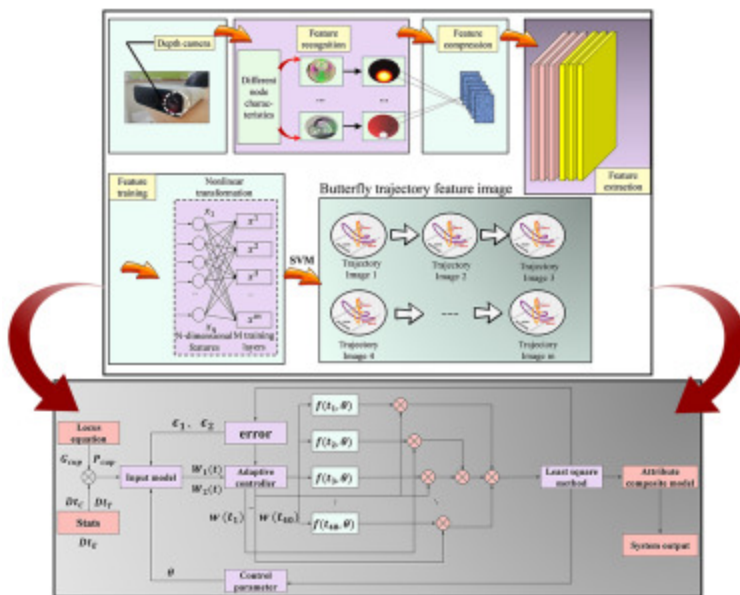
Help

## Summary

Implementing grasping tasks under color and multi scene promotion conditions is a key technology. This study proposes a recognition and grasping technique based on the crystal

butterfly algorithm and adaptive imitation synthesis. Firstly, inspired by the movement trajectory of butterflies, a dynamic node tracking method called "Butterfly Trajectory" was designed. It can complete dynamic trajectory tracking under geometric constraints and achieve route memory. The second color dynamic recognition technology (CDR) has been proposed. It can quickly extract brightness, transparency, and color saturation obtained from multiple angles. Improve the feature extraction speed of Region CNN (R-CNN) instead of traditional methods (HOG). In addition, an adaptive imitation synthesis technique (AISP) is used to achieve the multi scenario promotion of grasping technology. Finally, simulation and physical testing were provided to verify the effectiveness of the design scheme in this article.

## Graphical abstract



[Download: Download high-res image \(203KB\)](#)

[Download: Download full-size image](#)



Previous

Next

PDF

Help

Subject areas

## Introduction

In recent years, due to the increase in cargo transportation and the shortage of human resources, people have become increasingly interested in the research of gripping technology. It is an attractive solution to use visual or tactile recognition technology<sup>1,2,3</sup> as a "perception organ" to perform grasping tasks. This scheme has been widely verified in multiple fields.<sup>4,5,6,7</sup> This technology also has broad application scenarios in modern medicine. Such as medical rehabilitation robots,<sup>8</sup> automatic medicine-grabbing robots, and so forth.

The first article to employ recognition techniques in grasping technology seems to be [9].<sup>9</sup> Since then, people have been very interested in studying recognition techniques to accomplish grasping tasks. Hay and Chehadeh<sup>10</sup> proposed a noise-tolerant method based on a deep neural network (DNN). This method overcomes the rich domain knowledge required for system adjustment and can achieve real-time identification. Wang and Qiu<sup>11</sup> studied a transformer based on CNN architecture. Used to solve visual place recognition problems. The R-CNN designed in<sup>12</sup> uses the traditional CNN network instead of HOG to effectively improve the speed of feature extraction. It can quickly identify items to be grabbed among distractions. With the popularity of sensors, using them as identification mechanisms in grabbing has become another research area for scholars. Such as visual sensors,<sup>13</sup> proximity sensors,<sup>14</sup> and tactile sensors,<sup>15</sup> and so forth. In order to fuse and extract different recognition information obtained during grasping, Weng and Zhou<sup>16,17</sup> developed multi-modal datasets and multi-light sensing methods. They can provide a more comprehensive dataset for scraping. Along with the application of human-computer interaction, Li and Yin<sup>18</sup> proposed a visual and tactile fusion framework. This method improves the grasping success rate in complex environments by complementing visual and tactile data. In order to increase the application scenarios of,<sup>18</sup> Li S and Yu H<sup>19</sup> proposed a transparent object grasping technology based on the fusion of vision and touch. However, the promotion of this technology in complex background environments still needs to be improved. Jiang<sup>20</sup> designed a transparent object deep reconstruction and manipulation framework (A framework builds a large dataset (TRANS-AFF) to save object data and complete the detection of transparent objects.

It should be emphasized that most of the above model designs on recognition grasping focus on linear systems with fixed colors. Although<sup>20</sup> considered the effect of transparent

objects due to light reflection and refraction, it did not address the change of background color. [19]<sup>19</sup> focuses in color change, but the promotion of complex scenes is poor. It can be seen that the recognition capture of color changing objects brings new challenges to the existing techniques. Therefore it is imperative to study the recognition grasping of color changing objects.

Based on the above observations, we investigate a recognition grasping technique based on the Crystal Butterfly algorithm and adaptive imitation synthesis. The contribution of this article is 3-fold.

- 1) A dynamic node tracking method called "butterfly trajectory" is proposed in the Crystal Butterfly algorithm. It can accurately track the trajectory of color-changing object nodes and change the dynamic node dataset in real time. The trajectory detection and tracking effect is better than [21],<sup>21</sup> Compared with Dex-Nerf<sup>22</sup> and RGB-D,<sup>23</sup> it has richer data information.
- 2) A color dynamic recognition technology (CDR) is proposed in the Crystal Butterfly algorithm. To overcome the recognition bias caused by color changes,<sup>20,21,22,23,24</sup> this method incorporates the path memory of the butterfly trajectory. This makes it easier to extract multi-angle features of color-changing objects. Compared with [12]<sup>12</sup> and [19],<sup>19</sup> it has higher dynamic color recognition.
- 3) To expand the application of recognition and grasping in complex environments, we proposed the adaptive imitation synthesis technology (AISP). Specifically, this method overcomes the shortcomings of imitation learning,<sup>25</sup> such as poor data adaptability and simple copying, and focuses more on the adaptability of learning. It can continuously make new adjustments according to changes in environmental information and achieve multi-scenario grasping promotion.

The remainder of this article is organized as follows. Section II discusses the design process of the methodology regarding the dynamic node tracking of butterfly trajectories. Section III establishes the color dynamic recognition technique. Section IV discusses the establishment and application of adaptive mimicry synthesis. Section V gives simulations and physical tests to illustrate the theoretical results. Section VI summarizes this article. Section VII points out the limitations of the article and future research directions.

## Butterfly trajectory dynamic node tracking

The set of internal hierarchical nodes of a transparent object is represented as  $\mathbf{I}_{\text{cup}}$ . This set contains  $N$  nodes.  $\mathbf{I}_{\text{cup}} = \{(\mathbf{x}_{\text{in}1}, \mathbf{y}_{\text{in}1}, \mathbf{z}_{\text{in}1}), (\mathbf{x}_{\text{in}2}, \mathbf{y}_{\text{in}2}, \mathbf{z}_{\text{in}2}), \dots, (\mathbf{x}_{\text{in}N}, \mathbf{y}_{\text{in}N}, \mathbf{z}_{\text{in}N})\}$ . The set

of grabbing points when grabbing the glass is  $G_{cup}$ . It indicates where the grab point should be placed.  $G_{cup} = \{(x_{grip1}, y_{grip1}, z_{grip1}), (x_{grip2}, y_{grip2}, z_{grip2})\}$ . The center position of the transparent object is  $P_{cup} = \{x_{cup}, y_{cup}, z_{cup}\}$ . Both  $G_{cup}$  and  $P_{cup}$  are node collections at the external level. The internal hierarchical nodes and external hierarchical nodes are explained as follows.

- 1) Internal hierarchy nodes are key points of the internal geometry of transparent objects.
- 2) External hierarchy nodes are key points on the surface and edges of transparent objects.

These nodes are used to establish geometric relationships between each other. This allows the grasping device to adjust its posture based on the real-time position of the transparent object.

Select 5 key master nodes, represented by  $A_1, A_2, A_3, A_4$  and  $A_5$  respectively. Since the position of a node is a function of time. Therefore

$$I_{cup}(t) = \{ (x_{in1}(t), y_{in1}(t), z_{in1}(t)), (x_{in2}(t), y_{in2}(t), z_{in2}(t)), \dots, (x_{inN}(t), y_{inN}(t), z_{inN}(t)) \},$$

$$G_{cup}(t) = \{(x_{grip1}(t), y_{grip1}(t), z_{grip1}(t)), (x_{grip2}(t), y_{grip2}(t), z_{grip2}(t)), (x_{inN}(t), y_{inN}(t), z_{inN}(t)) \}.$$

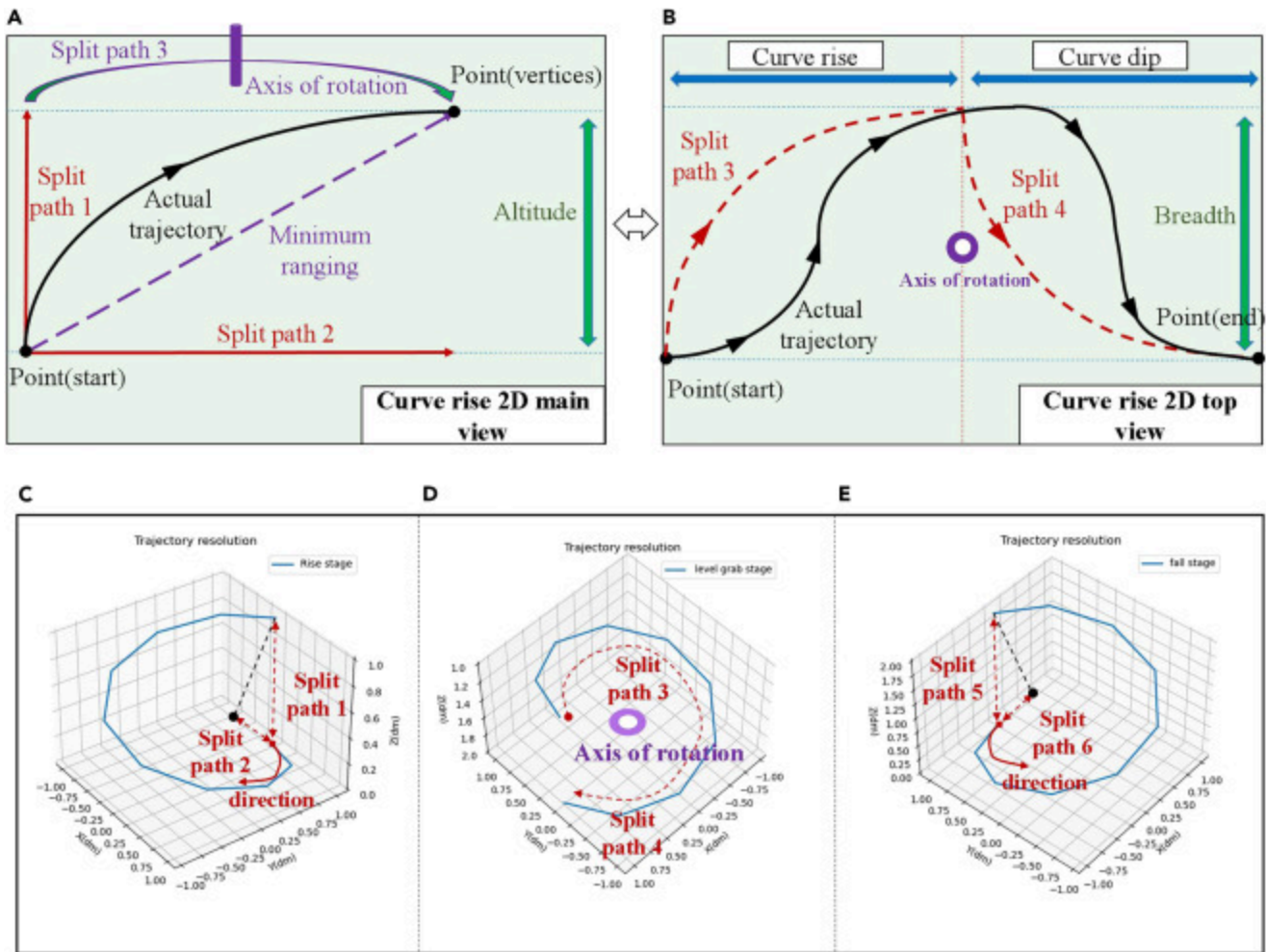
$$P_{cup}(t) = \{x_{cup}(t), y_{cup}(t), z_{cup}(t)\}.$$

All the grabbing processes will go through grabbing (depalletizing), curve rising, curve descending and placing (palletizing). Therefore  $G_{cup}$  will change from the initial position before grabbing to the final position.

$$G_{cup}(t_0) = \{(x_{grip1}(t_0), y_{grip1}(t_0), z_{grip1}(t_0)), (x_{grip2}(t_0), y_{grip2}(t_0), z_{grip2}(t_0))\} \rightarrow G_{cup}(t_2) = \{(x_{grip1}(t_2), y_{grip1}(t_2), z_{grip1}(t_2)), (x_{grip2}(t_2), y_{grip2}(t_2), z_{grip2}(t_2))\}$$

$t_0$  represents the start time of grasping, and  $t_2$  represents the end time of grasping. The grasping trajectory is divided into different stages, as shown in [Figure 1](#).

- 1) Rising stage.



[Download: Download high-res image \(928KB\)](#)

[Download: Download full-size image](#)

Figure 1. Trajectory split

(A) is the main view trajectory route.

(B) is the top view trajectory route.

(C-E) is the trajectory route and separation path at different grabbing stages.

The center point ( $P_{cup}$ ) of the transparent object changes as it rises.

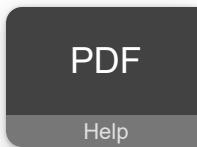
$$P_{cup}(t) = \{x_{cup}(t), y_{cup}(t), z_{cup}(t)\}. z_{cup}(t) \text{ will increase.}$$

2) Horizontal rotation stage.

The position and direction of the grab point set ( $G_{cup}$ ) change at the same time.

$$G_{cup}(t) = \{ (x_{grip1}(t), y_{grip1}(t), z_{grip1}(t)), (x_{grip2}(t), y_{grip2}(t), z_{grip2}(t)) \} \text{ and } x_{grip2}(t) \text{ will}$$

$$x_{grip1}(t)$$



change.

3) Descending stage.

The grab point sets  $G_{cup}$  and  $P_{cup}$  change with descent.

$$G_{cup}(t) = \{z_{grip1}(t), z_{grip2}(t)\} \text{ and } \{(x_{grip1}(t), y_{grip1}(t), z_{grip1}(t)), (x_{grip2}(t), y_{grip2}(t), z_{grip2}(t))\},$$

$$P_{cup}(t) = \{x_{cup}(t), y_{cup}(t), z_{cup}(t)\}$$

$z_{cup}(t)$  will decrease.

Based on the above information, the trajectory equation of  $G_{cup}$  is,

$$G_{cup}(t) \quad \text{(Equation 1)}$$

$$= \begin{cases} (x_{start1}, y_{start1}, z_{start1} + v_{up1}(t - t_0)), & t_0 < t < t_1 \\ (x_{rotatel1}(t - t_1), y_{rotatel1}(t - t_1), z_{rotatel1}(t - t_1)), & t_1 < t < t_2 \\ (x_{down1}, y_{down1}, z_{down1} - v_{down1}(t - t_2)), & t > t_2 \end{cases}$$

where  $t_0$ ,  $t_1$  and  $t_2$  represent the time points when the grabbing occurs, the rotation occurs and the end time respectively.

$x_{start1}$ ,  $y_{start1}$ ,  $z_{start1}$ ,  $x_{rotatel1}$ ,  $y_{rotatel1}$ ,  $z_{rotatel1}$ ,  $x_{down1}$ ,  $y_{down1}$ ,  $z_{down1}$  respectively represent the coordinate positions of three time nodes.  $v_{up1}$  and  $v_{down1}$  represent the rising and falling speeds.

The trajectory equations of  $P_{cup}(t)$  and  $I_{cup}(t)$  both satisfy,

$$P_{cup}(t) \quad \text{(Equation 2)}$$

$$= \begin{cases} (x_{start2} + f(t), y_{start2}, z_{start2}), & t_0 < t < t_1 \\ (x_{rotatel2} + f(t_1) \cos(g(t)), y_{rotatel2} + f(t_1) \sin(g(t)), z_{rotatel2}) & t_1 < t < t_2 \\ (x_{down2} + f(t_1) \cos(g(t_2)) + h(t), y_{down2} + f(t_1) \sin(g(t_2))), & t > t_2 \end{cases}$$

where  $x_{start2}$ ,  $y_{start2}$ ,  $z_{start2}$ ,  $x_{rotatel2}$ ,  $y_{rotatel2}$ ,  $z_{rotatel2}$ ,  $x_{down2}$ ,  $y_{down2}$ ,  $z_{down2}$  respectively represent the coordinate positions of three time nodes.  $f(t)$  is the rising trajectory function.  $g(t)$  is the descending trajectory function.  $h(t)$  is the rotation function.  $f(t_1)$  and  $g(t_2)$  are the rising trajectory function and falling trajectory function at  $t_1$  and  $t_2$  moments, respectively. The motion trajectory of the glass grabbing point satisfies (1). The motion trajectories of the five key nodes all satisfy (2). Consider picking an arbitrary point from keypoints and grab points. Take  $A_1$  as an example. Trajectory tracking via trajectory equations. Complete route memory for transparent objects. Assume  $A_1 = (x, y, z)$ ,

establish a second-order motion model,

$$\mathbf{A}_1(t) = [\mathbf{x}(t), \mathbf{y}(t), \mathbf{z}(t), \dot{\mathbf{x}}(t), \dot{\mathbf{y}}(t), \dot{\mathbf{z}}(t), \ddot{\mathbf{x}}(t), \ddot{\mathbf{y}}(t), \ddot{\mathbf{z}}(t)]^T \quad (\text{Equation 3})$$

$\mathbf{A}_1(t)$  is the status vector. This formula contains the position information, velocity information, and acceleration information during the movement of the node. According to the second-order motion model, the state transition equation is,

$$\mathbf{A}_1(t) = \mathbf{B}\mathbf{A}_1(t-1) + \mathbf{C}\mathbf{U}(t) + \delta_t \quad (\text{Equation 4})$$

where  $\mathbf{B}$  is the state transition matrix.  $\mathbf{C}$  is the control input matrix.  $\mathbf{U}(t)$  is the control input vector.  $\delta_t$  is the process noise in the state transition equation.

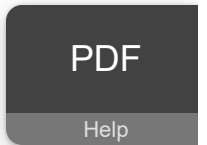
If the different stages of trajectory splitting are regarded as uniform linear motion, then the state transition matrix at this time can be expressed as,

$$\mathbf{B} = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 & \frac{1}{2}(\Delta t)^2 & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 & 0 & \frac{1}{2}(\Delta t)^2 & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t & 0 & 0 & \frac{1}{2}(\Delta t)^2 \\ 0 & 0 & 0 & 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (\text{Equation 5})$$

where  $\Delta t$  is the time interval between adjacent stages in the grasping process.

The control input matrix is expressed as,

$$\mathbf{C} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad (\text{Equation 6})$$





This control matrix can realize speed control of  $A_1$  in the three directions of  $\dot{x}(t)$ ,  $\dot{y}(t)$  and  $\dot{z}(t)$  during constant speed rise, horizontal constant speed rotation, and constant speed descent. However, there is an actual curved motion during the grabbing process. Its trajectory direction and velocity magnitude are constantly changing. Therefore, the state transition matrix and control input matrix for the curve motion trajectory become,

$$B = \begin{bmatrix} 1 & 0 & \Delta t & 0 & \frac{1}{2}(\Delta t)^2 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \Delta t & 0 & \frac{1}{2}(\Delta t)^2 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & \Delta t & 0 & \frac{1}{2}(\Delta t)^2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \Delta t & 0 & \frac{1}{2}(\Delta t)^2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & \Delta t & 0 & \frac{1}{2}(\Delta t)^2 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (\text{Equation 7})$$

$$C = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad (\text{Equation 8})$$

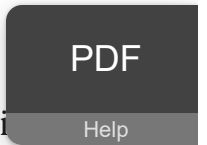
During the trajectory tracking process, the observation equation is set to,

$$O(t) = PA_1(t) + \varepsilon_t \quad (\text{Equation 9})$$

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (\text{Equation 10})$$

where  $P$  is the observation matrix.  $\varepsilon_t$  is the observation noise.  $O(t)$  is the observation vector.

During the observation process, the state transition equation and observation equation are updated as,



$$\widehat{\mathbf{A}}_1(t) = \mathbf{B}\widehat{\mathbf{A}}_1(t-1) + \mathbf{C}\mathbf{U}(t) + \delta_t \quad (\text{Equation 11})$$

$$\widehat{\mathbf{O}}(t) = \mathbf{P}\widehat{\mathbf{A}}_1(t) + \varepsilon_t \quad (\text{Equation 12})$$

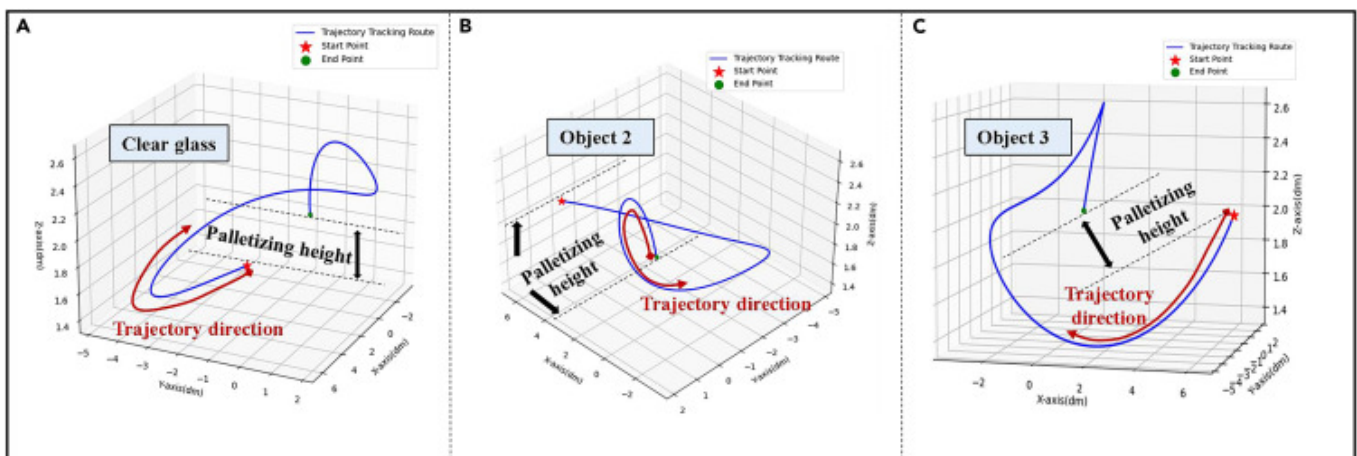
where  $\widehat{\mathbf{A}}_1(t)$  is the updated state transition function at time  $t$ .  $\widehat{\mathbf{A}}_1(t-1)$  is the state transition function at time  $t-1$  before the update.  $\widehat{\mathbf{O}}(t)$  is the updated observation function at time  $t$ .

According to the trajectory equation motion model, the state transition equation and the observation equation are estimated and predicted.

$$\widehat{\mathbf{A}}_1(t) = \mathbf{B}\widehat{\mathbf{A}}_1(t-1) + \mathbf{G}_{\text{cup}}(t) [\mathbf{P}\mathbf{C}\mathbf{U}(t) + \varepsilon_t] \quad (\text{Equation 13})$$

$$\widehat{\mathbf{O}}(t) = \mathbf{P}\widehat{\mathbf{A}}_1(t) + \mathbf{P}_{\text{cup}}(t) [\mathbf{P}\mathbf{C}\mathbf{U}(t) + \delta_t] \quad (\text{Equation 14})$$

Based on the above observations, [Figure 2](#) shows the trajectories of different objects. [Figure 3](#) is the dynamic debugging process for different numbers of trajectory routes in the update of state transition equations and observation equations. It can be seen from [Figures 2](#) and [3](#) that the stacking height and trajectory direction of different objects are different. Therefore, the trajectory of each object is different.



[Download: Download high-res image \(593KB\)](#)

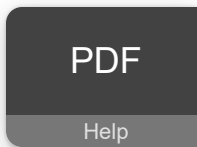
[Download: Download full-size image](#)

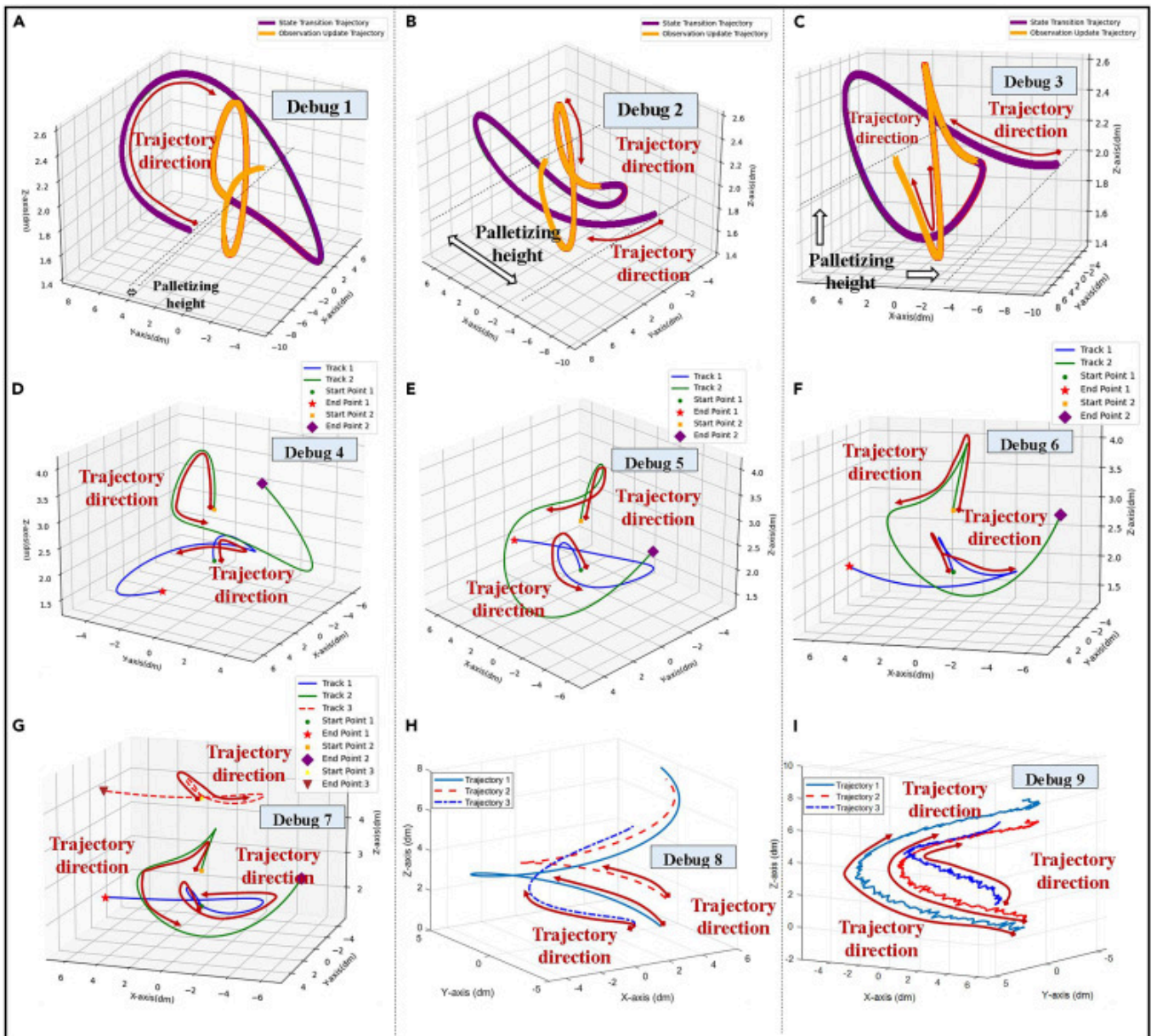
Figure 2. The trajectory of different items

(A) is the trajectory of a transparent glass.

(B) is the trajectory of a non-transparent object.

(C) is the trajectory of another non-transparent object.





[Download: Download high-res image \(2MB\)](#)

[Download: Download full-size image](#)

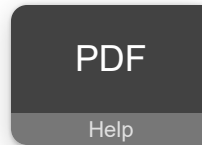
Figure 3. Dynamic debugging of trajectory route

(A–C) is the dynamic trajectory of one key node.

(D–F) is the dynamic trajectory of two key nodes.

(G–I) is the dynamic trajectory of three key nodes.

**Remark 1:** The butterfly trajectory dynamic node tracking method uses the second-order motion model and state transition matrix to describe the motion process of transparent objects. In this method, the internal hierarchical node set and the external hierarchical node set of the transparent object are set, and these nodes represent the key feature points of the



object. The grasping trajectory is split into different stages. The trajectory equations and observation equations of different nodes collect rich motion data compared to [22–23].<sup>22,23</sup> The estimated prediction of the state transition equation combined with the control input can be used to perform the dynamic trajectory tracking of key nodes in different stages. It is worth noting that the dynamic node tracking method is only suitable for grasping tasks of transparent items of specific shapes. It is not suitable when the grabbing environment is interfered with by non-transparent items with the same shape and different colors as the grabbing target. Therefore, it is necessary to establish color dynamic recognition technology (CDR) based on this method.

## Color dynamic recognition technology (CDR)

CDR abandons the feature extraction method of traditional R-CNN.<sup>12</sup> Its key ingredients are the fusion of multimodal data into a comprehensive dataset and diversified feature analysis.

According to the dynamic trajectory,  $n$  action photos of transparent objects are captured through the depth camera. The depth camera model used in the experimental test is Kinect DK. Kinect DK can make full use of its depth sensing and RGB image capture capabilities. Especially when dealing with transparent objects, Kinect DK's depth sensor can help reduce visual interference caused by transparency and reflection, thereby improving the accuracy of object recognition. For the convenience of testing, let  $n = 40$ . The temporal collection of action photos is represented as  $\mathbf{Time} = \{t_1, t_2, t_3, \dots, t_{40}\}$ . Get the brightness, transparency, and color saturation of transparent objects for each action photo. Represents  $\mathbf{Lum} = \{l_1, l_2, l_3, \dots, l_{40}\}$ ,  $\mathbf{Par} = \{p_1, p_2, p_3, \dots, p_{40}\}$ ,  $\mathbf{Col} =$  respectively. Time is  $\{c_1, c_2, c_3, \dots, c_{40}\}$  expressed as  $\mathbf{Time} = \{t_1, t_2, t_3, \dots, t_{40}\}$ . Remove missing values from three datasets.

$$\begin{cases} \mathbf{Dt}_{d1} = \mathbf{Lum}. \text{dropna} () \\ \mathbf{Dt}_{d2} = \mathbf{Par}. \text{dropna} () \\ \mathbf{Dt}_{d3} = \mathbf{Col}. \text{dropna} () \end{cases} \quad (\text{Equation 15})$$

where  $\text{dropna}()$  is a function in the Pandas library. Its role is to remove missing values from the data.  $\mathbf{Dt}_{d1}$ ,  $\mathbf{Dt}_{d2}$  and  $\mathbf{Dt}_{d3}$  are the datasets after deleting missing values, respectively.

After removing missing values from rough data to prevent clustering, the deleted are filled with data so that the data still maintains the original number of samples.

$$\begin{cases} \mathbf{Dt}_{d1} = \mathbf{Dt}_{d1}. \text{fillna} () \\ \mathbf{Dt}_{d2} = \mathbf{Dt}_{d2}. \text{fillna} () \\ \mathbf{Dt}_{d3} = \mathbf{Dt}_{d3}. \text{fillna} () \end{cases} \quad (\text{Equation 16})$$

PDF

Help

where fillna() is a function in the Pandas library, which is used to fill in missing values in the data. Data filling of  $Dt_{d1}$ ,  $Dt_{d2}$  and  $Dt_{d3}$  realizes data update. The sample data is normalized after cleaning.

$$\begin{cases} Dt_{s1} = \frac{Dt_{d1} - \min(Dt_{d1})}{\max(Dt_{d1}) - \min(Dt_{d1})} \\ Dt_{s2} = \frac{Dt_{d2} - \min(Dt_{d2})}{\max(Dt_{d2}) - \min(Dt_{d2})} \\ Dt_{s3} = \frac{Dt_{d3} - \min(Dt_{d3})}{\max(Dt_{d3}) - \min(Dt_{d3})} \end{cases} \quad (\text{Equation 17})$$

where min() and max() are the minimum and maximum samples in the dataset respectively, and  $Dt_{s1}$ ,  $Dt_{s2}$  and  $Dt_{s3}$  are the normalized datasets respectively. Data are compressed into the range [0, 1].

Analytical hierarchy process (AHP)<sup>26</sup> is a quantitative analysis method for multi-criteria decision-making problems. In this article, it hierarchizes the preprocessed data of transparent objects and determines the relative weight of each factor by constructing a comparison matrix. You can use the weight distribution formula:

$$W_A \cdot A + W_B \cdot B + W_C \cdot C \quad (\text{Equation 18})$$

where A, B and C represent any sample among  $Dt_{s1}$ ,  $Dt_{s2}$  and  $Dt_{s3}$ .  $W_A$ ,  $W_B$  and  $W_C$  are the weights of A, B, and C, respectively, which satisfy  $W_A + W_B + W_C = 1$ . The purpose is to ensure the standardization of weights. Construct a comparison matrix ( $P_{ij}$ ).

$$P_{ij} = \begin{bmatrix} 1 & \frac{A}{B} & \frac{A}{C} \\ \frac{B}{A} & 1 & \frac{B}{C} \\ \frac{C}{A} & \frac{C}{B} & 1 \end{bmatrix} \quad (\text{Equation 19})$$

Perform eigenvalue decomposition on the comparison matrix and obtain the weight vector as

$$P_{ij} \cdot \alpha = \lambda \cdot \alpha \quad (\text{Equation 20})$$

where  $\alpha$  is the eigenvector corresponding to each eigenvalue. Standardize  $\alpha$  to obtain the final weight vector ( $\beta$ ). The process of normalization is to divide each component of the feature vector by the sum of all its components to ensure  $W_A + W_B + W_C = 1$ .

$$\text{Result} = \beta_A \cdot A + \beta_B \cdot B + \beta_C \cdot C \quad (\text{Equation 21})$$

where **Result** is the final output result.  $\beta_A$ ,  $\beta_B$ , and  $\beta_C$  are the final weight vectors.





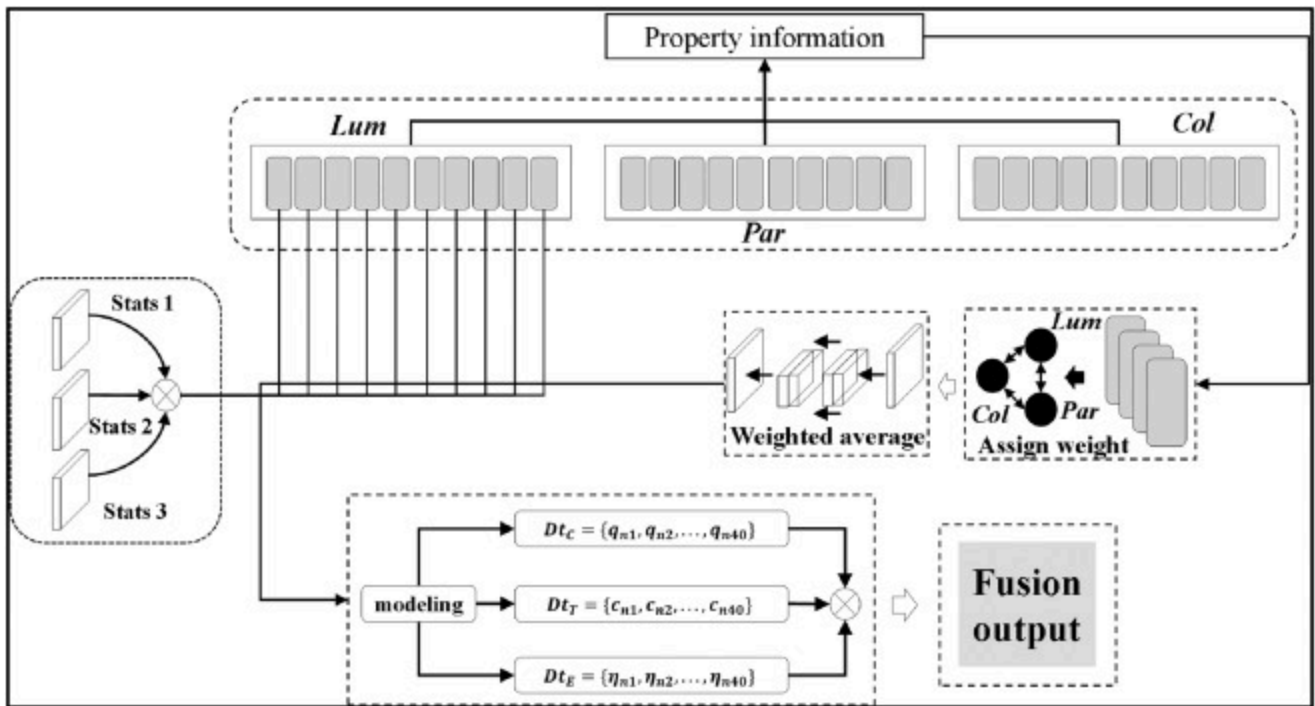
**Remark 2:** This article uses AHP to allocate weights to achieve hierarchical processing. Not only can it give a reasonable relative weight, but it can also reflect the actual influence of each factor in decision-making analysis. It effectively avoids complex information similar to TRANS-AFF.<sup>20</sup>

Perform weighted averaging on chromaticity information.

$$\text{Avg} = \frac{\beta_A \cdot A + \beta_B \cdot B + \beta_C \cdot C}{\beta_A + \beta_B + \beta_C} \quad (\text{Equation 22})$$

Construct a multi-modal dataset as shown in Figure 4 to complete image training. The new multimodal dataset is denoted

$$Dt_C = \{q_{n1}, q_{n2}, q_{n3}, \dots, q_{n40}\}, Dt_T = \{c_{n1}, c_{n2}, c_{n3}, \dots, c_{n40}\}, Dt_E = \{\eta_{n1}, \eta_{n2}, \eta_{n3}, \dots, \eta_{n40}\}$$



[Download: Download high-res image \(399KB\)](#)

[Download: Download full-size image](#)

Figure 4. Multimodal dataset design architecture

The multi-modal datasets are fused and a feature image with a size of  $100 \times 100$  pixels is output. A feature matrix is built to represent different samples and features respectively. Map the values in the matrix to pixel values to generate an image. Use the image processing library Matplotlib to create images that represent the values in the matrix as colors. The obtained feature images are introduced into the R-CNN model to extract color change

information of transparent objects. When training the R-CNN model, choosing the right hyperparameters has a crucial impact on the performance of the model. The key training hyperparameters used in this article include the following.

The learning rate is set to 0.001, and an adaptive learning rate adjustment algorithm is used. This allows the learning rate to be adjusted dynamically during training, ensuring that the model learns quickly in the early stages and adjusts finely in the later stages. The batch size is set to 32. The number of training rounds is set to 100, and the performance of the validation set determines the optimal number of training rounds to prevent overfitting or underfitting. The momentum is set to 0.9. By introducing the momentum term, we are able to reach the optimal solution faster during training. The weight decay is 0.0005. It helps control the model complexity and ensures that the model does not remember the noise in the training set during training. The cross entropy loss is used as the loss function for the classification task. The formula is as follows:

$$L(\mathbf{y}, \hat{\mathbf{y}}) = - \sum_{i=1}^n y_i \log(\hat{y}_i) \quad (\text{Equation 23})$$

where  $\mathbf{y}$  is the true label and  $\hat{\mathbf{y}}$  is the predicted probability.

The regression loss is used to adjust the regression of the bounding box, and the calculation formula is:

$$L_{\text{reg}}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n \text{smooth}_{L1}(\mathbf{x}_i - \mathbf{y}_i) \quad (\text{Equation 24})$$

where  $\text{smooth}_{L1}$  is a loss function that linearizes larger errors and squares smaller errors.

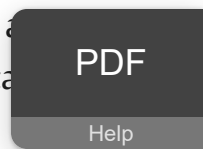
The feature image at this time is the feature output of each attribute information. The selective search algorithm can obtain candidate areas of feature images. The target candidate area is set to 500, and the specific process is as follows.

- 1) Initialize the set of candidate regions and equate the image to the initial candidate region. Similarity calculation is performed by color similarity measure and region size similarity measure.

Select the area with the lowest similarity for segmentation to generate a candidate area. Repeat this step until the target candidate area is 500 and stop the operation. The candidate area information is represented as  $\text{region} = \{\mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_{500}\}$ .

The color similarity measure and the region size similarity measure are expressed as follows.

$$S_{\text{color}}(\mathbf{k}_i, \mathbf{k}_j) = \sum_{\mathbf{m}=\mathbf{r}_1}^{\mathbf{r}_{500}} \min(H_i(\mathbf{m}), H_j(\mathbf{m})) \quad (\text{Equation 25})$$



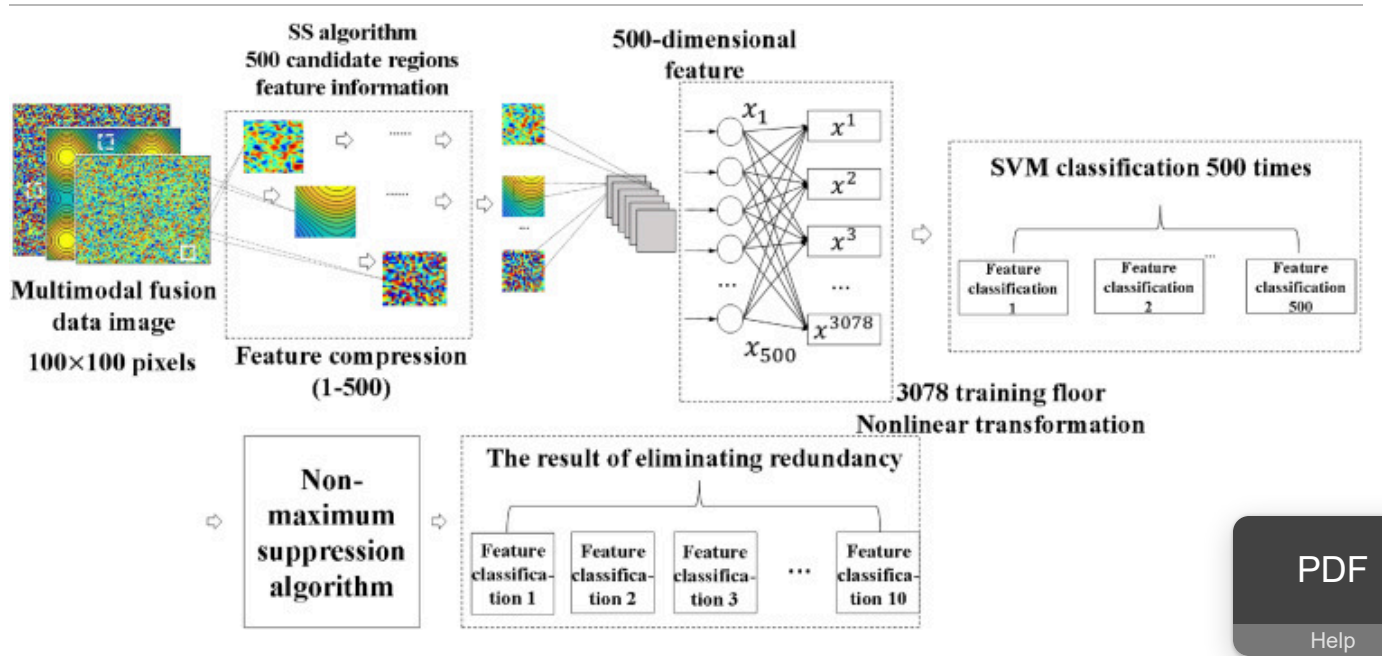
$$S_{\text{size}}(\mathbf{k}_i, \mathbf{k}_j) = 1 - \frac{|\text{size}(\mathbf{k}_i) - \text{size}(\mathbf{k}_j)|}{\max(\text{size}(\mathbf{k}_i), \text{size}(\mathbf{k}_j))} \quad (\text{Equation 26})$$

where  $\mathbf{k}_i$  and  $\mathbf{k}_j$  are different neighborhoods of any candidate area.  $\mathbf{H}_i(\mathbf{m})$  and  $\mathbf{H}_j(\mathbf{m})$  are the comparison rates of  $\mathbf{k}_i$  and  $\mathbf{k}_j$  in the  $m$ -th candidate area.  $\text{size}(\mathbf{k}_i)$  and  $\text{size}(\mathbf{k}_j)$  are the neighborhood areas of  $\mathbf{k}_i$  and  $\mathbf{k}_j$ .

Candidate region compression uses fixed ratio compression. Set up 50 classifiers, 500-dimensional features, and 3078 training layers.

$$\begin{bmatrix} x_1^1 & x_2^1 & \dots & x_{3078}^1 \\ x_1^2 & x_2^2 & \dots & x_{3078}^2 \\ \dots & \dots & \dots & \dots \\ x_1^{500} & x_2^{500} & \dots & x_{3078}^{500} \end{bmatrix} \begin{bmatrix} \omega_1^1 & \dots & \omega_{50}^1 \\ \omega_2^1 & \dots & \omega_{50}^2 \\ \dots & \dots & \dots \\ \omega_{3078}^1 & \dots & \omega_{50}^{3078} \end{bmatrix} \quad (\text{Equation 27})$$

The SVM classification criterion is adopted in the multi-modal fusion image model training in Figure 5. And use the non-maximum suppression algorithm (Non-maximum Suppression) to eliminate repeated features after SVM classification. The candidate area generated by the selective search algorithm is the area with the smallest similarity among the three performance indicators. That is, the candidate area where the color change is most likely to occur. Select 10 feature samples after training in the candidate area, and perform similar superposition, close comparison, and similarity value analysis on them, as shown in Table 1.



Download: [Download high-res image \(773KB\)](#)

Download: [Download full-size image](#)

Figure 5. Multi-modal fusion image model

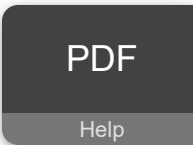




Table 1. Candidate feature data processing

Data analysis	Different moments nodes										
Luminance	0.51	0.78	0.88	0.83	0.97	0.98	0.88	0.69	0.41	0.2	
Transparency	0.24	0.27	0.38	0.42	0.5	0.59	0.68	0.79	0.88	1	
Saturation	0.39	0.58	0.63	0.67	0.78	0.89	0.93	0.98	0.99	1	
Superposition of similarities	0.24	0.4	0.5	0.32	0.38	0.18	0.1	0.58	1.16	1.6	
Draw a close comparison	0.42	0.43	0.39	0.96	1.11	1.92	2.19	0.72	1.2	2.6	
Similarity Analysis 1	0.599		0.0354		0.1878		0.0366		0.0976		
Similarity Analysis 2		0.319		0.4746		0.0366		0.2202			

**Remark 3:** The key to this step is to use a similar feature extraction method to monitor color changes by detecting whether the properties of transparent objects deviate from the normal range. This idea makes monitoring more flexible and effective and provides a new method for capturing transparent objects, thereby realizing the judgment of color change information of transparent objects.

### Adaptive imitation synthesis (AISP)

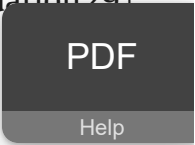
The three key time nodes for secondary grabbing are set as  $t_s$ ,  $t_m$  and  $t_e$ . They respectively represent the grabbing start time, the rotation occurrence time and the grabbing end time. The dynamic trajectory equation of the entire process can be expressed as,

$$P(t) = f(t, t_s, t_m, t_e) \tag{Equation 28}$$

where  $P(t)$  is the real-time location of the grabbing device.  $f$  is the motion characteristic function. According to (1) and (2), the dynamic trajectory of the secondary grabbing node set is established.

$$G_{cup}(t) = \begin{cases} (x_{start1}, y_{start1}, z_{start1} + v_{up1}(t - t_s)), & t_s < t < t_m \\ (x_{rotatell}(t - t_m), y_{rotatell}(t - t_m), z_{rotatell}(t - t_m)), & t_m < t < t_e \\ (x_{down1}, y_{down1}, z_{down1} - v_{down1}(t - t_e)), & t > t_e \end{cases} \tag{Equation 29}$$

where  $v_{up1}$  and  $v_{down1}$  represent the rising and falling speeds.



The dynamic trajectory tracking equations of  $P_{cup}$  and  $I_{cup}$  satisfy,

(Equation30)

$$P_{cup}(t) / I_{cup} = \begin{cases} (x_{start2} + f(t), y_{start2}, z_{start2}), & t_s < t < t_m \\ (x_{rotatel2} + f(t_m) \cos(g(t)), y_{rotatel2} + f(t_m) \sin(g(t)), z_{rotatel2}), & t_m < t < t_e \\ (x_{down2} + f(t_m) \cos(f(t_e)) + h(t), y_{down2} + f(t_m) \sin(f(t_e)), z_{down2}), & t > t_e \end{cases}$$

$g(t)$  is the rotation function.  $h(t)$  is the excitation function.  $f(t_m)$  and  $f(t_e)$  are the rising trajectory function and falling trajectory function at  $t_m$  and  $t_e$  moments respectively.

Establish a mapping relationship between dynamic trajectory tracking equations and object attribute information in time series to form a comprehensive dataset (Data).

$$\text{Data} = \begin{cases} G_{cup}(t_1) = (x_1, y_1, z_1, \dot{x}_1, \dot{y}_1, \dot{z}_1), & t = t_1 \\ \dots\dots\dots, & \dots\dots\dots \\ G_{cup}(t_{40}) = (x_{40}, y_{40}, z_{40}, \dot{x}_{40}, \dot{y}_{40}, \dot{z}_{40}), & t = t_{40} \\ P_{cup}(t_1) = (x_1, y_1, z_1, \dot{x}_1, \dot{y}_1, \dot{z}_1), & t = t_1 \\ \dots\dots\dots, & \dots\dots\dots \\ P_{cup}(t_{40}) = (x_{40}, y_{40}, z_{40}, \dot{x}_{40}, \dot{y}_{40}, \dot{z}_{40}), & t = t_{40} \\ (\text{Lum}(t_1), \text{Par}(t_1), \text{Col}(t_1)), & t = t_1 \\ \dots\dots\dots, & \dots\dots\dots \\ (\text{Lum}(t_{40}), \text{Par}(t_{40}), \text{Col}(t_{40})), & t = t_{40} \end{cases} \quad (\text{Equation31})$$

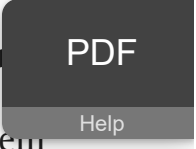
where  $x_i, y_i, z_i, \dot{x}_i, \dot{y}_i, \dot{z}_i$  are the trajectory position and speed at time  $t=i$  respectively.

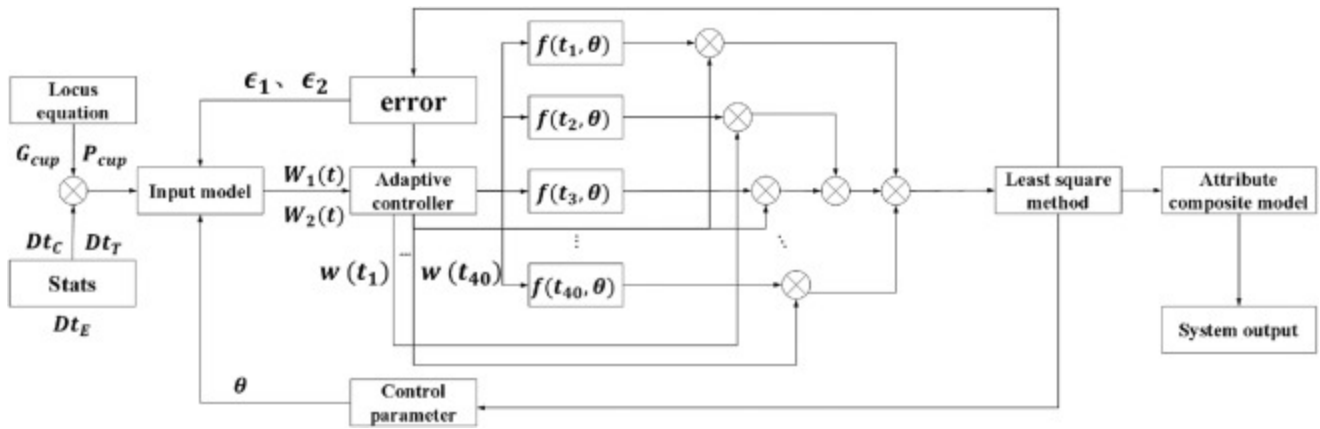
Design an adaptive control system, as shown in Figure6. The system is trained using the established comprehensive dataset. The control parameters are determined by learning the information association between dynamic trajectory information and object attributes. Set up the system debugging function during the capture process and establish a system debugging model in the form of the system debugging function.

$$W_1(t) = G_{cup}(t) + f(t, \theta) + \epsilon_1 \quad (\text{Equation32})$$

$$W_2(t) = P_{cup}(t) + f(t, \theta) + \epsilon_2 \quad (\text{Equation33})$$

where  $W_1(t)$  is the system debugging function of the grab point.  $W_2(t)$  is the system debugging function of the central point and internal key nodes.  $f(t, \theta)$  is the dynamic model of the control parameter  $\theta$  at time  $t$ . Both  $\epsilon_1$  and  $\epsilon_2$  are model errors. Trajectory tracking equations and attribute information serve as inputs to the adaptive control system.





[Download: Download high-res image \(176KB\)](#)

[Download: Download full-size image](#)

Figure6. Adaptive control system

According to the dynamic trajectory dataset,  $\theta$ ,  $\epsilon_1$  and  $\epsilon_2$  are determined by the least squares method.  $\theta$ ,  $\epsilon_1$  and  $\epsilon_2$  real-time feedback input model. Establish the time correspondence between system debugging functions:

$$W_{t1} = \{(t_1, W_1(t_1)), (t_2, W_1(t_2)), (t_3, W_1(t_3)), \dots, (t_{40}, W_1(t_{40}))\}.$$

$$W_{t2} = \{(t_1, W_2(t_1)), (t_2, W_2(t_2)), (t_3, W_2(t_3)), \dots, (t_{40}, W_2(t_{40}))\}.$$

$$W(t) = \{(t_1, W(t_1)), (t_2, W(t_2)), (t_3, W(t_3)), \dots, (t_{40}, W(t_{40}))\}$$

A loss function is introduced to minimize the objective function through the sum of squares of the residuals.

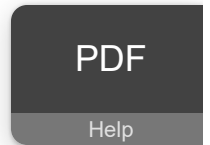
$$\begin{cases} J(\theta, \epsilon_1) = \sum_{i=1}^{40} [W_1(t) - f(t_i, \theta) - \epsilon_1]^2 \\ J(\theta, \epsilon_2) = \sum_{i=1}^{40} [W_2(t) - f(t_i, \theta) - \epsilon_2]^2 \end{cases} \quad (\text{Equation 34})$$

Let the partial derivative of the minimized objective function be equal to zero.

$$\begin{cases} \partial(J(\theta, \epsilon_1)) / \partial\theta = 0 \\ \partial(J(\theta, \epsilon_1)) / \partial\epsilon_1 = 0 \\ \partial(J(\theta, \epsilon_2)) / \partial\theta = 0 \\ \partial(J(\theta, \epsilon_2)) / \partial\epsilon_2 = 0 \end{cases} \quad (\text{Equation 35})$$

The normal equation of the least squares method can be obtained through (8):

$$Y^T Y \begin{bmatrix} \theta \\ \epsilon_1 \\ \epsilon_2 \end{bmatrix} = Y^T S \quad (\text{Equation 36})$$



where the size of  $\mathbf{Y}$  is a  $3 \times 3$  design matrix. It contains the relevant values of  $\theta$ .  $\mathbf{S}$  is the information vector of object attributes of  $\mathbf{Dt}_C(t_i)$ ,  $\mathbf{Dt}_T(t_i)$  and  $\mathbf{Dt}_E(t_i)$  at  $t_i$ .

Based on the above information, the parameters are continuously adjusted with the help of the least squares method.  $\theta$ ,  $\epsilon_1$  and  $\epsilon_2$  are finally determined. Establish a transparent object attribute model based on the determined parameters.

$$\begin{cases} \mathbf{B}(t) = \mathbf{q}_k \cdot e^{-\alpha t} \\ \mathbf{T}(t) = \mathbf{c}_k \cdot e^{-\beta t} \\ \mathbf{S}(t) = \boldsymbol{\eta}_k \cdot e^{-\gamma t} \end{cases} \quad (\text{Equation37})$$

where  $\mathbf{q}_k$  is any brightness sample in  $\mathbf{Dt}_C = \{q_{n1}, q_{n2}, q_{n3}, \dots, q_{n40}\}$ .  $\alpha$  is the attenuation coefficient related to optical properties and ambient lighting.  $\mathbf{B}(t)$  represents the time change function of brightness.  $\mathbf{c}_k$  is any transparency sample in  $\mathbf{Dt}_T = \{c_{n1}, c_{n2}, c_{n3}, \dots, c_{n40}\}$ .  $\beta$  is the transparency attenuation coefficient.  $\mathbf{T}(t)$  represents the time change function of transparency.  $\boldsymbol{\eta}_k$  is any color saturation sample in  $\mathbf{Dt}_E = \{\eta_{n1}, \eta_{n2}, \eta_{n3}, \dots, \eta_{n40}\}$ .  $\gamma$  is the color saturation attenuation coefficient.  $\mathbf{S}(t)$  represents the time change function of color saturation. The trajectory and attribute composite model of a transparent object are as follows.

$$\begin{cases} \mathbf{F}_1(\mathbf{B}, \mathbf{T}, \mathbf{S}, \lambda) = \mathbf{B}(t) \mathbf{W}_1(t) \cdot \mathbf{T}(t) \mathbf{W}_1(t) \\ \quad \cdot \mathbf{S}(t) \mathbf{W}_1(t) + \mathbf{W}_2(t) \cdot \lambda \\ \mathbf{F}_2(\mathbf{B}, \mathbf{T}, \mathbf{S}, \lambda) = \mathbf{B}(t) \mathbf{W}_2(t) \cdot \mathbf{T}(t) \mathbf{W}_2(t) \\ \quad \cdot \mathbf{S}(t) \mathbf{W}_2(t) + \mathbf{W}_1(t) \cdot \lambda \\ \lambda = \frac{\alpha}{\beta} + \frac{\beta}{\gamma} + \frac{\gamma}{\alpha} \end{cases} \quad (\text{Equation38})$$

where  $\mathbf{F}_1(\mathbf{B}, \mathbf{T}, \mathbf{S}, \lambda)$  and  $\mathbf{F}_2(\mathbf{B}, \mathbf{T}, \mathbf{S}, \lambda)$  are composite functions about the trajectory and attribute model of transparent objects.  $\lambda$  is a constant.

Based on the above information, the adaptive control system state vector is established.

$$\mathbf{Vec}(t) = [\mathbf{F}_1(\mathbf{B}, \mathbf{T}, \mathbf{S}, \lambda), \mathbf{F}_2(\mathbf{B}, \mathbf{T}, \mathbf{S}, \lambda)] \quad (\text{Equation39})$$

We design an adaptive controller.

$$\boldsymbol{\nu}(t) = \mathbf{K}_p \cdot \mathbf{e}(\epsilon_1, \epsilon_2) + \mathbf{K}_d \cdot \mathbf{Vec}(t) + \mathbf{K}_i \cdot \int \mathbf{e}(\epsilon_1, \epsilon_2) dt \quad (\text{Equation40})$$

where  $\boldsymbol{\nu}(t)$  is the controller output.  $\mathbf{e}(\epsilon_1, \epsilon_2)$  is the dynamic trajectory error.  $\mathbf{K}_p$  is the proportional gain. It determines the proportional relationship between  $\boldsymbol{\nu}(t)$  and  $\mathbf{e}(\epsilon_1, \epsilon_2)$ .  $\mathbf{K}_d$  is the differential gain. It determines the response relationship between  $\boldsymbol{\nu}(t)$  and

PDF

Help

$\text{Vec}(t)$ .  $\mathbf{K}_i$  is the integral gain. It reflects the integral relationship between  $\nu(t)$  and  $e(\epsilon_1, \epsilon_2)$  and has the function of eliminating steady-state errors.

According to (15),  $\mathbf{K}_i \cdot \int e(\epsilon_1, \epsilon_2) dt$  is expanded and the saturation function  $\mathbf{O}(x)$  is introduced. It is used to control the change growth of  $\mathbf{K}_i \cdot \int e(\epsilon_1, \epsilon_2) dt$ .

$$\nu(t) = \mathbf{K}_p \cdot e(\epsilon_1, \epsilon_2) + \mathbf{K}_d \cdot \text{Vec}(t) + \int \mathbf{O}(\mathbf{K}_i \cdot \int e(\epsilon_1, \epsilon_2) dt) dt \quad (\text{Equation41})$$

Adaptive gains  $\mathbf{K}_\alpha, \mathbf{K}_\beta$  and  $\mathbf{K}_\gamma$  are introduced in the adaptive adjustment mechanism. It can complete the online state adjustment in transparent object grasping. The adaptive gain relationship of proportional gain, differential gain and integral gain changing with time is established.

$$\begin{cases} \mathbf{K}_p(t) = \mathbf{K}_\alpha \cdot \mathbf{K}_{p0} \\ \mathbf{K}_d(t) = \mathbf{K}_\beta \cdot \mathbf{K}_{d0} \\ \mathbf{K}_i(t) = \mathbf{K}_\gamma \cdot \mathbf{K}_{i0} \end{cases} \quad (\text{Equation42})$$

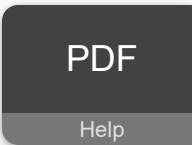
where  $\mathbf{K}_{p0}, \mathbf{K}_{d0}$  and  $\mathbf{K}_{i0}$  are the gain coefficients before the time change. Based on the above information, the design process of the adaptive control system is shown in [Table 2](#).

Table 2. Adaptive control system design process

---

**Input:**  $G_{cup}(t), P_{cup}(t), Dt_C, Dt_T, Dt_E, Vec(t), v(t), W(t), e(\epsilon_1, \epsilon_2)$   
 Learning rate(learning\_rate), Number of iterations(I), Maximum Iterations(MI), Allowance(TE), Control parameter( $\theta$ )

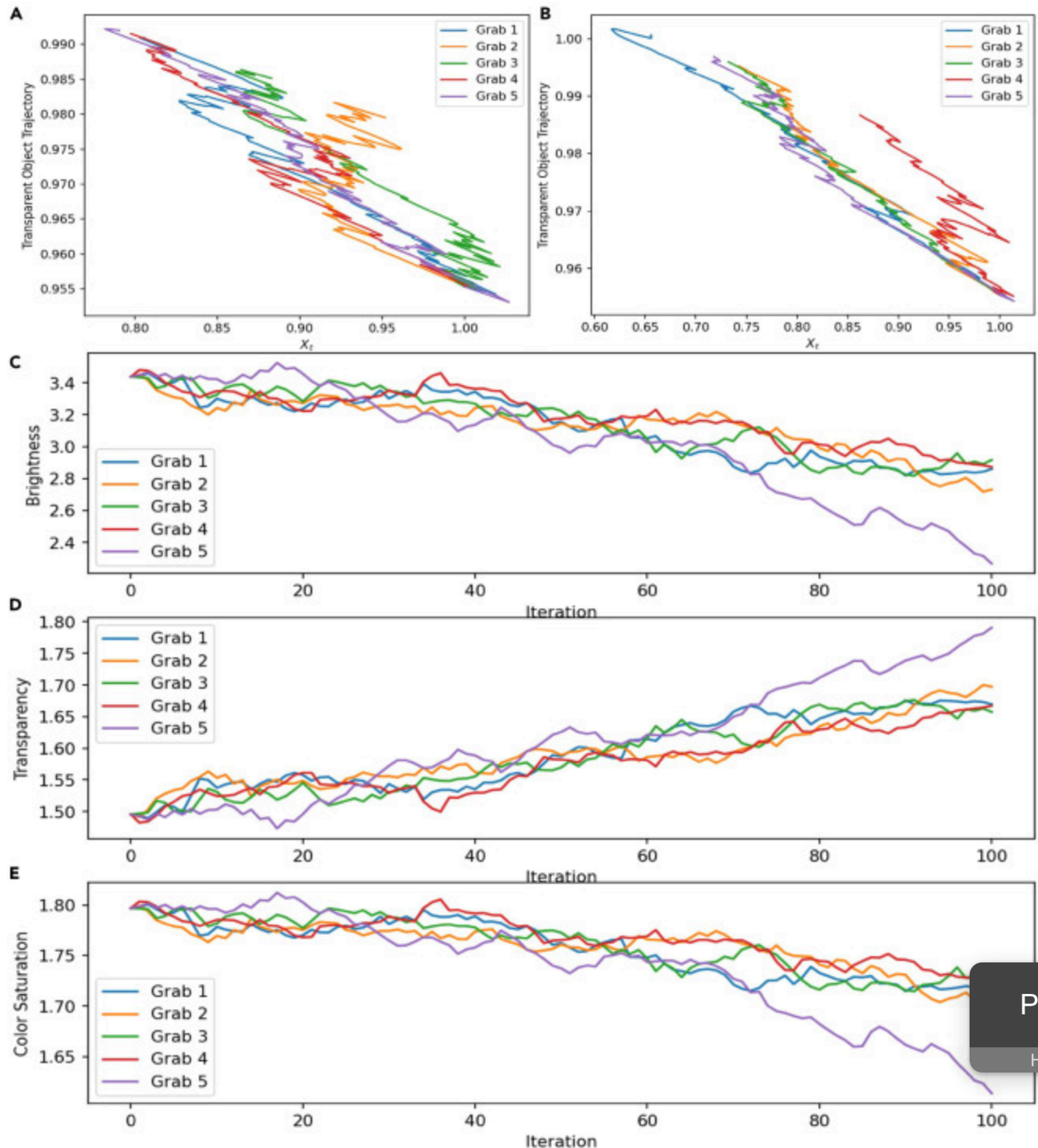
Step1:  $G_{cup}(t) = \text{compute\_trajectory}(\text{Para})$   
 Step2:  $P_{cup}(t) = \text{compute\_trajectory\_properties}(G_{cup}(t))$   
 Step3:  $Dt_C, Dt_T, Dt_E = \text{compute\_attributes}(P_{cup}(t))$   
 Step4:  $I = 1 : e(\epsilon_1, \epsilon_2) = e(\epsilon_1, \epsilon_2) - \text{function}(G_{cup}(t), P_{cup}(t))$   
 Step5: Calculated gradient:  $\text{Grad} = \text{Gradient}(e(\epsilon_1, \epsilon_2), W(t))$   
 Step6: Update parameter:  $\theta = \theta - \text{learning\_rate} * \text{grad}$   
 Step7: Calculate the new dynamic trajectory error:  $\text{new\_}e(\epsilon_1, \epsilon_2) = e(\epsilon_1, \epsilon_2) - \text{function}(\theta)$   
 Step8: if( $|\text{new\_}e(\epsilon_1, \epsilon_2)| < \text{TE}$ )  
 $I = I + 1$   
 $Vec(t) = [G_{cup}(t), P_{cup}(t), Dt_C, Dt_T, Dt_E]$   
 $v(t) = \text{ompute\_contro\_output}(\text{optimized\_}\theta, Vec(t))$   
 Step9: if( $I > \text{MI}$ )



Output optimal control parameters  $\rightarrow \theta$

Out of the loop

In the adjustment parameters of the adaptive control system, the iterative process of grabbing trajectories and attribute information for different transparent objects is shown in Figure 7.



[Download: Download high-res image \(1MB\)](#)

[Download: Download full-size image](#)

PDF

Help

Figure 7. Iteration of transparent object trajectory and attribute information

(A and B) is the trajectory iteration of 5 grabs.

(C–E) is the attribute iteration of 5 grabs.

Here, a total of five grabbing processes from Grab1 to Grab5 are simulated. During the iteration process of the transparent object trajectory, the horizontal axis represents the discrete steps of time. The total simulation time is 1 min, divided into 1000 steps. Trajectory iterations gradually converge to a fit with discrete steps. Reflects the process of optimizing parameter adjustment. The error of object attribute information in 100 iterations is within a small range.

It can be seen from Equations 2 and 3 that  $G_{cup}$ ,  $P_{cup}$  and  $I_{cup}$  are three-dimensional trajectories parameterized with respect to time. That is

$G_{cup}(t) = [x(t), y(t), z(t)]$ ,  $P_{cup}(t) = [x(t), y(t), z(t)]$ ,  $I_{cup}(t) =$  Establish a  $[x(t), y(t), z(t)]$ .

relationship with  $\theta$ . That is  $G_{cup}(t, \theta)$ ,  $P_{cup}(t, \theta)$ ,  $I_{cup}(t, \theta)$ . Establish a relationship between the three.

$$G_{cup}(t, \theta) = \underset{\theta}{\operatorname{argmin}} \sum_{para=1}^{MI} \| P_{cup}(Dt_C[para], \theta), I_{cup}(Dt_T[para], \theta), Dt_E(para, \theta) \| \quad (\text{Equation 43})$$

where  $para$  is a constant.  $Dt_C[para] = q_{para-1}$ ,  $Dt_T[para] = c_{para-1}$ ,  $Dt_E[para] = \eta_{para-1}$ .  $Dt_E(para, \theta)$  represents the color saturation of  $\eta_{para-1}$  related to the control parameter.

The changes in attributes over time and control parameters during learning can be continuously adjusted according to (18). The adjustment process is as follows.

$$Dt_C(t, \theta) = \frac{1}{n} \sum_{para=1}^{40} \left( \frac{\partial G_{cup}(t, \theta)}{\partial t} \right)^2 \quad (\text{Equation 44})$$

$$Dt_T(t, \theta) = \exp\left(- \int_{para=1}^{40} \frac{\partial G_{cup}(t, \theta)}{\partial \theta}\right) \quad (\text{Equation 45})$$

$$Dt_E(t, \theta) = \sqrt{\sum_{para=1}^{40} \left( \frac{\partial G_{cup}(t, \theta)}{\partial t} \right)^2 + \left( \frac{\partial G_{cup}(t, \theta)}{\partial \theta} \right)^2} \quad (\text{Equation 46})$$

At this time, the system debugging function  $W(t)$  can be used to quantify the dynamic trajectory error  $e(\epsilon_1, \epsilon_2)$ .

$$W(t, \theta) = \sum_{\text{para}=1}^{40} (G_{\text{cup}}(t, \theta) - e(\epsilon_1, \epsilon_2))^2 \quad (\text{Equation 47})$$

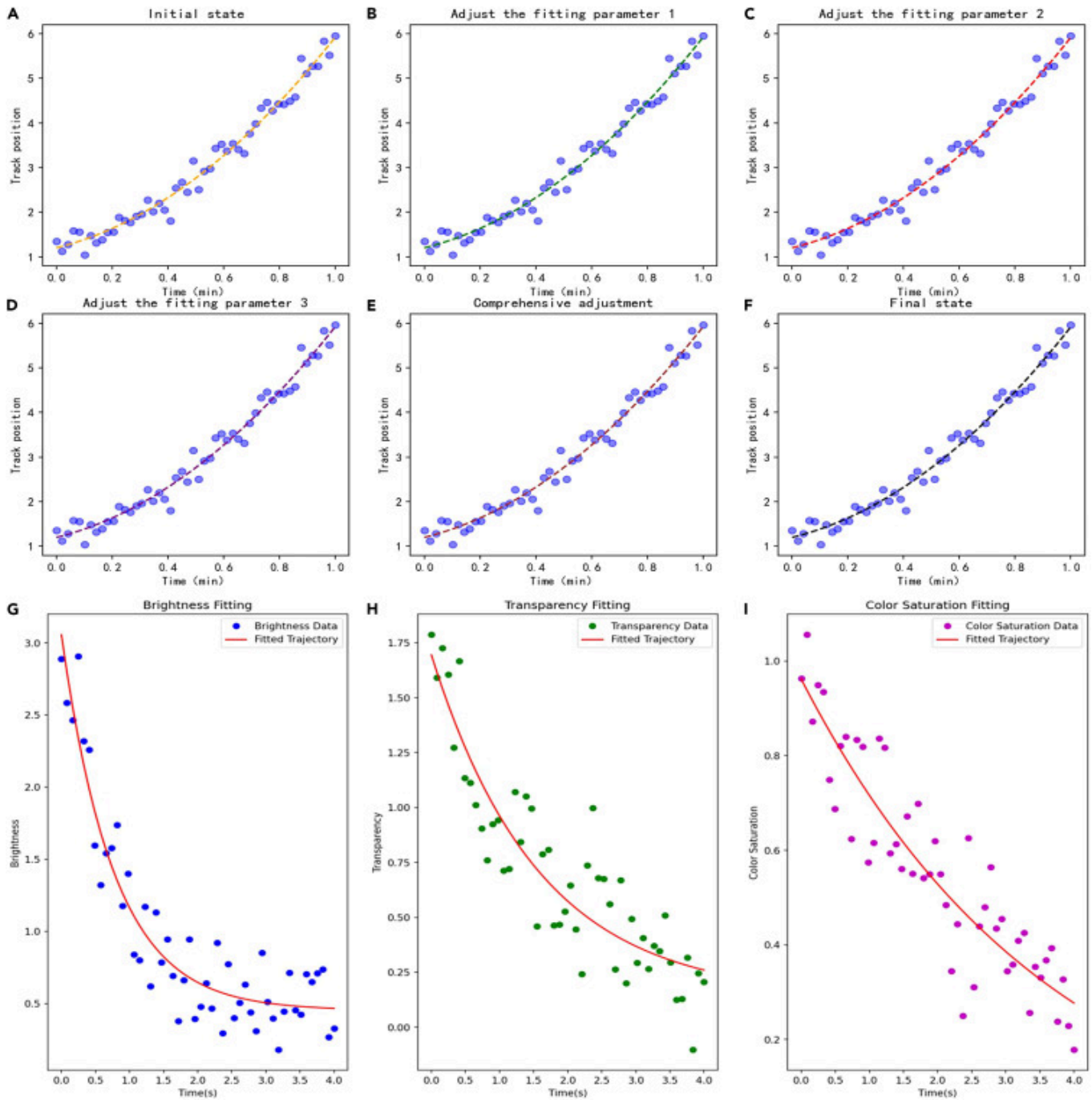
$$+ (P_{\text{cup}}(t) - e(\epsilon_1, \epsilon_2))^2 + (I_{\text{cup}}(t) - e(\epsilon_1, \epsilon_2))^2$$

Then the state vector  $\text{Vec}(t)$  in the adaptive control system synthesizes the above data information.

$$\text{Vec}(t) = \begin{bmatrix} G_{\text{cup}}(t, \theta) \\ Dt_C(t, \theta) \\ Dt_T(t, \theta) \\ Dt_E(t, \theta) \end{bmatrix} \quad (\text{Equation 48})$$

**Figure 8** averages multiple capture data of trajectories and attributes through adaptive imitation learning. The fitting effect is shown.





[Download: Download high-res image \(840KB\)](#)

[Download: Download full-size image](#)

Figure 8. Fitting effect after averaging the trajectory and attribute data

(A–F) is the trajectory fitting effect under different parameter states.

(G–I) is the attribute fitting effect under different parameter states.

Figures 8A–8F shows that as the parameters are adjusted, the trajectory fitting effect becomes more and more concentrated. This shows that the model can better capture the

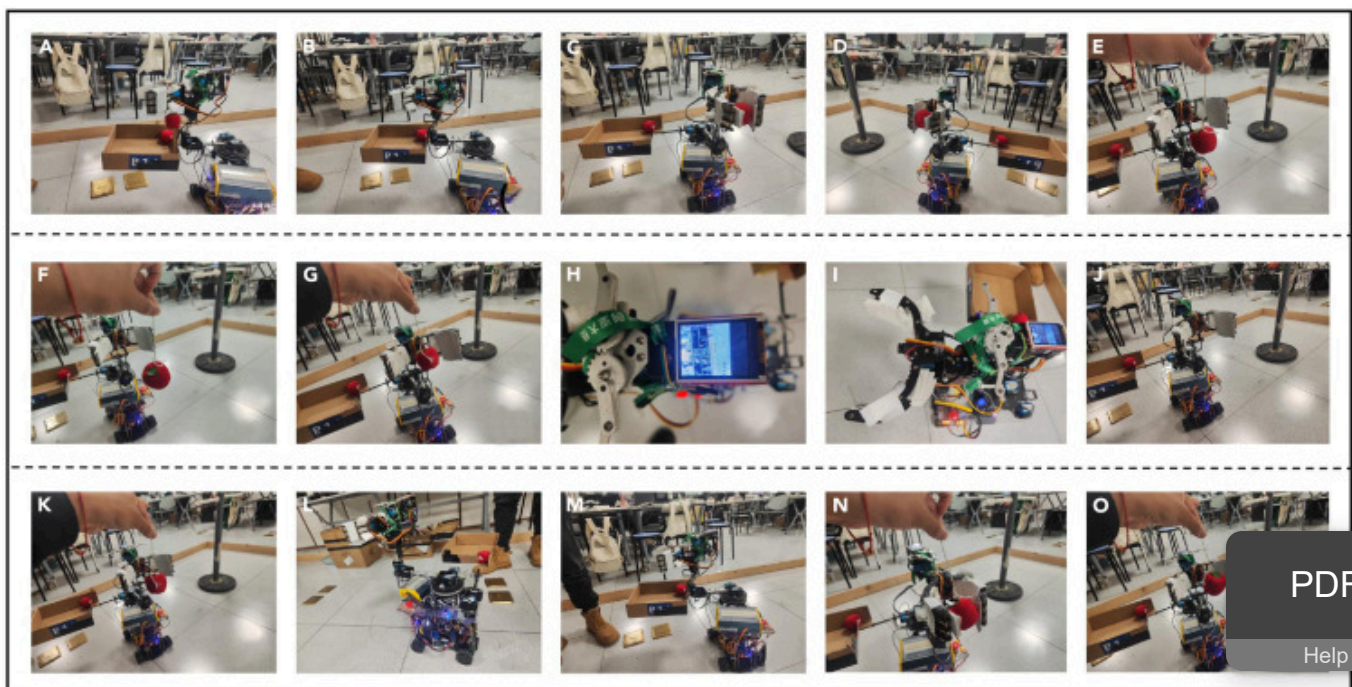
PDF  
Help

data characteristics and patterns of the trajectory. The fitting effect in [Figures 8G–8I](#) also confirms this.

## Results

The proposed technology is compared with two traditional grasping technologies. They are the lightweight robot grasping technology based on template matching depth image<sup>27</sup> (TMDI) and the grasping technology based on visual recognition processing<sup>28</sup> (VRP).

Since the two methods cannot be directly compared with the proposed method, we considered the more mainstream generative grasping network for comparison. The grasping comparison test is implemented with the help of this generative grasping network. A transparent glass cup is a typical color-changing object. Its color is different under different lighting or background. Since the front-end design of the grasping robot in the laboratory belongs to the "soft robot," it is more suitable for grasping soft transparent objects. Due to the characteristics of the transparent glass cup itself and the limitations of the laboratory lighting test environment, we use colored balls of the same shape but different colors to replace the transparent glass cup. These balls of different colors can be equivalent to different color changes of the same transparent object. Their color changes are more obvious, which is conducive to experimental testing, as shown in [Figure 9](#).



[Download: Download high-res image \(1MB\)](#)

[Download: Download full-size image](#)

Figure9. Physical grabbing test

(A–E) is the grabbing test at angle 1.

(F–J) is the grabbing test at angle 2.

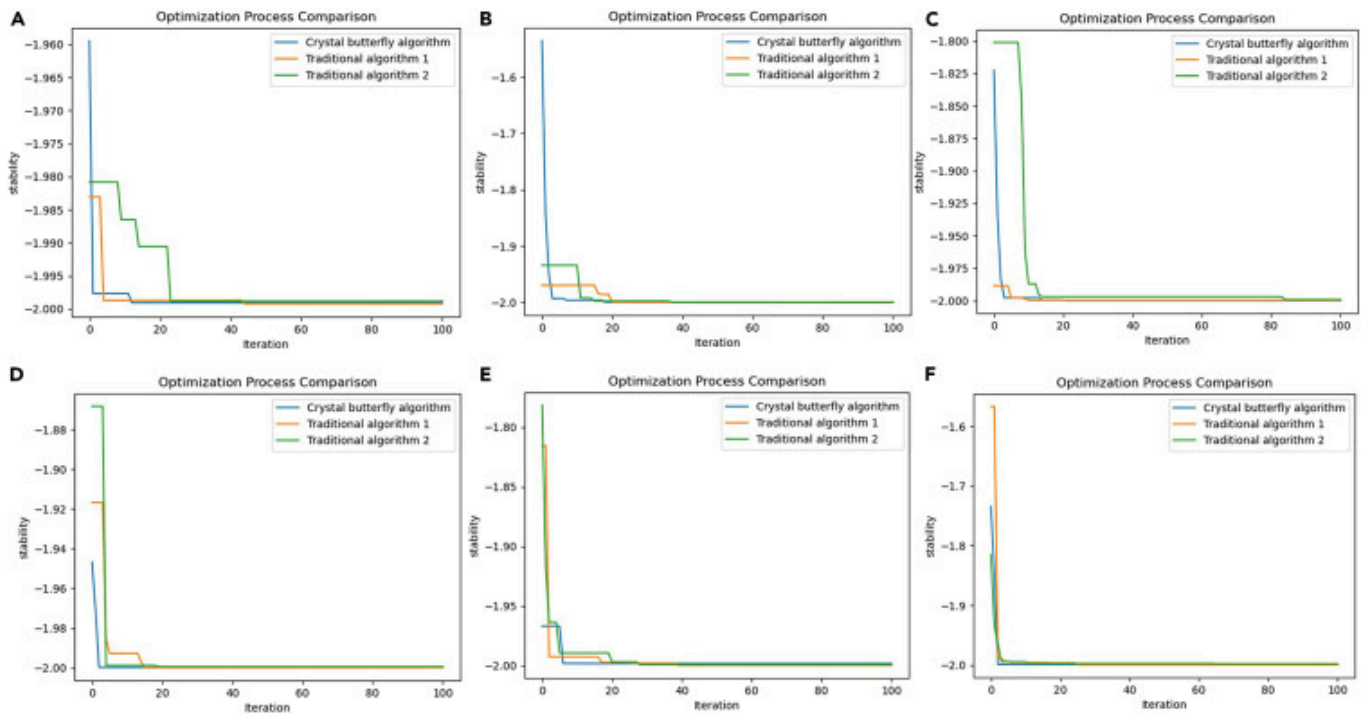
(K–O) is the grabbing test at angle 3. The grabbing environment at different angles is different.

---

Set the total number of grasps to 40. Consider five basic scenarios.

- 1) Scenario 1: only contains the object to be grasped, no interference information, and no color change.
- 2) Scenario 2: There is interference information, but no color change.
- 3) Scenario 3: There is interference information and there is color change.
- 4) Scenario 4: The interference information is more complex and there is no color change.
- 5) Scenario 5: The interference information is more complex and there is color change.

**Figure 10** shows the test process. We compared the success rate and average cycle of 40 grasping, as shown in **Tables 3** and **4**. The tabular data shows that in the grasping environment without color change, the grasping success rate of the three methods is relatively high. VPR and the method in this article both reached 92.5%. The grasping success rate of TMID in scenario 2 is higher than that in scenario 1, indicating the instability of the system. Once color changes and complex interference occur, the grasping success rates of TMID and VPR drop significantly. However, the method in this article still maintains a relatively high grasping success rate. Its average grasping success rate is stable at 86.5%. The average grasping cycle is stable at 18.66s.



[Download: Download high-res image \(455KB\)](#)

[Download: Download full-size image](#)

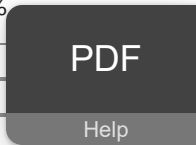
Figure 10. Stability comparison of different algorithms

(A–F) are the results of multiple tests on the stability of TMDI, VRP and the algorithm proposed in this article.

Table 3. Comparison of grasping success rates across 5 scenarios

Scene	The number of grabs $N=40$				
	Scene 1	Scene 2	Scene 3	Scene 4	Scene 5
TMDI	85%	90%	55%	75%	22.5%
VPR	92.5%	87.5%	40%	82.5%	42.5%
This method	92.5%	90%	85%	87.5%	77.5%

Table 4. Comparison of average grasping cycle times across 5 scenarios



Scene	The number of grabs $N=40$				
	Scene 1	Scene 2	Scene 3	Scene 4	Scene 5
TMDI	17.61s	23.18s	22.68s	24.32s	23.59s
VPR	21.47s	20.57s	24.29s	24.70s	27.46s
This method	10.03s	16.92s	19.66s	23.04s	23.65s

We tested and compared the algorithm stability during grabbing. A stable grasping algorithm can produce consistent results in different scenarios and initial conditions. This consistency is crucial for application in practical environments. TMDI and VPR are sensitive to initial conditions and are prone to fall into local optimal solutions in certain scenarios, resulting in the stagnation of grasping accuracy for a period of time. The Crystal Butterfly algorithm proposed in this article is more likely to stabilize during the iteration process, and different test results are shown in [Figure 10](#). The number of iterations of TMDI and VPR is mostly concentrated around 20 times or even more than 20 times. The number of iterations of the method in this article is less than 20 times. Multiple test results also confirm this point.

## Discussion

This article studies a crystal butterfly algorithm and adaptive imitation synthesis recognition and grabbing technology. The proposed butterfly trajectory dynamic node tracking method achieves accurate tracking and real-time adjustment of transparent object trajectories, effectively improving the trajectory detection and tracking effect. Combined with color dynamic recognition technology, it successfully overcomes the recognition bias caused by color changes. And improve the accuracy and efficiency of multi-angle feature extraction. Finally, adaptive imitation synthesis technology is introduced to make up for the shortcomings of traditional imitation learning such as poor data adaptability and simple copying, and realize the promotion of multi-scenario grabbing applications.

## Limitations of the study

The grasping device in this article is a single agent. In practical applications, collaborative applications between multiple agents are often involved. When a transparent body requires the collaborative operation of multiple machines, multi-dimensional and complex nonlinear terms will be involved. For these nonlinear terms, the least squares identification scheme is an interesting test. Designing relevant estimators and controllers based on the least squares

identification scheme to achieve the global stability of the nonlinear system is a feasible solution. In the future, we plan to extend this technology to the application scenario of multi-agent collaborative grasping. And design a new least squares identification scheme and apply it to more complex multi-dimensional nonlinear systems to achieve collaborative control.

## STAR★Methods

### Key resources table

REAGENT or RESOURCE	SOURCE	IDENTIFIER
<b>Software and algorithms</b>		
PyCharm Community Edition 2023.2.5	Python Software Foundation	<a href="https://www.python.org">https://www.python.org</a> ↗
MATLAB R2022a	Matlab Software Foundation	<a href="https://ww2.mathworks.cn/products/matlab-home.html">https://ww2.mathworks.cn/products/matlab-home.html</a> ↗
Selective Search algorithm	This manuscript	N/A
Butterfly Optimization Algorithm	This manuscript	N/A
Attribute Supply Material 1	<a href="https://data.mendeley.com/datasets/x9yw8k77vd/1">https://data.mendeley.com/datasets/x9yw8k77vd/1</a> ↗	<a href="https://doi.org/10.17632/x9yw8k77vd.1">https://doi.org/10.17632/x9yw8k77vd.1</a> ↗
Attribute Supply Material 2	<a href="https://data.mendeley.com/datasets/x9yw8k77vd/1">https://data.mendeley.com/datasets/x9yw8k77vd/1</a> ↗	<a href="https://doi.org/10.17632/x9yw8k77vd.1">https://doi.org/10.17632/x9yw8k77vd.1</a> ↗
Track information supply materials	<a href="https://data.mendeley.com/datasets/x9yw8k77vd/1">https://data.mendeley.com/datasets/x9yw8k77vd/1</a> ↗	<a href="https://doi.org/10.17632/x9yw8k77vd.1">https://doi.org/10.17632/x9yw8k77vd.1</a> ↗
Test Code	<a href="https://zenodo.org/records/12542227">https://zenodo.org/records/12542227</a> ↗	<a href="https://doi.org/10.5281/zenodo.12542227">https://doi.org/10.5281/zenodo.12542227</a> ↗
<b>Other</b>		
Stacking and unstacking equipment	Anqiu Boyang Machinery Manufacturing Co., Ltd	<a href="http://www.sdbyjx.net">http://www.sdbyjx.net</a> ↗





REAGENT or RESOURCE	SOURCE	IDENTIFIER
Grab device	Robotics R&D Laboratory of Shandong Technology and Business University	N/A
Financial support	Shandong Provincial Technology Innovation Guidance Program	YDZX2023030
Depth camera	Kinect DK	N/A
R-CNN Model	This manuscript	N/A
CDR	This manuscript	N/A
Multi-modal fusion image model	This manuscript	N/A

## Resource availability

### Lead contact

Further information and requests for resources and reagents should be directed to and will be fulfilled by the lead contact, Chuanyu Cui ([cuichuanyu1999@163.com](mailto:cuichuanyu1999@163.com) ↗).

### Materials availability

This study did not generate new unique reagents.

### Data and code availability

- In the grabbing test of dismantling and stacking, the route data of butterfly trajectory and transparent object attribute data have been deposited at Mendeley Data, which are publicly available as of the date of publication. The DOI is listed in the [key resources table](#).
- All original code has been deposited at Zenodo and is publicly available as of the publication. DOIs are listed in the [key resources table](#).
- Any additional information required to reanalyze the data reported in this paper is available from the [lead contact](#) upon request.

## Experimental model and study participant details

PDF

Help

## Intelligent grasping experiment

The experimental part of the grasping test in this study was carried out in the Robot Control Laboratory of Shandong Technology and Business University and verified by Anqiu Boyang Machinery Manufacturing Co., Ltd.

**Model Design:** This study introduces a recognition and grasping technique for color-variable objects based on the Crystal Butterfly Algorithm and Adaptive Imitation Synthesis. The primary experimental subject is a transparent glass, with colored balls used to simulate color changes under different lighting and background conditions. The experimental setup includes a depth camera (Kinect DK) and a soft robotic gripper.

**Data Acquisition:** Data on the brightness, transparency, and color saturation of transparent objects were collected using the Kinect DK depth camera. Specific experimental parameters are as follows:

Number of Photos: 40.

Time set:  $\mathbf{Time} = \{t_1, t_2, \dots, t_{40}\}$ . Brightness Data Set:  $\mathbf{Lum} = \{l_1, l_2, \dots, l_{40}\}$ .

Transparency Data Set:  $\mathbf{Par} = \{p_1, p_2, \dots, p_{40}\}$ . Color Saturation Data Set:

$\mathbf{Col} = \{c_1, c_2, \dots, c_{40}\}$ .

**Study Subjects:** The primary subject of the study is a transparent glass, with colored balls simulating color changes under various lighting and background conditions to test the grasping success rate and cycle in different scenarios.

**Experimental Scenarios:**

Scenario 1: Only the target object is present, no interference, no color change.

Scenario 2: Interference is present, but no color change.

Scenario 3: Interference and color change are present.

Scenario 4: More complex interference, no color change.

Scenario 5: More complex interference and color change.

**Data Processing and Analysis:**

**Data Cleaning:** Missing values are deleted and filled.

**Normalization:** Data are scaled to a range of [0, 1].

PDF

Help



Feature Extraction: Using the Analytical Hierarchy Process (AHP) to assign weights and merge multimodal data sets.

Model Training: Features are trained using the R-CNN model, and classified using the SVM algorithm.

Experimental Results:

Grasping Success Rate and Cycle Comparison: Results are shown in [Tables 3](#) and [4](#).

In 40 grasping tests, the proposed method maintained an average grasping success rate of 86.5% and an average cycle of 18.66 seconds across different scenarios.

## Method details

### R-CNN model

The learning rate is set to 0.001, and an adaptive learning rate adjustment algorithm is used. This allows the learning rate to be adjusted dynamically during training, ensuring that the model learns quickly in the early stages and adjusts finely in the later stages. The batch size is set to 32. The number of training rounds is set to 100, and the performance of the validation set determines the optimal number of training rounds to prevent overfitting or underfitting. The momentum is set to 0.9. By introducing the momentum term, we are able to reach the optimal solution faster during training. The weight decay is 0.0005. It helps control the model complexity and ensures that the model does not remember the noise in the training set during training. The cross entropy loss is used as the loss function for the classification task.  $L(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_{i=1}^n y_i \log(\hat{y}_i)$ , where  $\mathbf{y}$  is the true label and  $\hat{\mathbf{y}}$  is the predicted probability. The regression loss is used to adjust the regression of the bounding box, and the calculation formula is  $L_{\text{reg}}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n \text{smooth}_{L1}(\mathbf{x}_i - \mathbf{y}_i)$ , where  $\text{smooth}_{L1}$  is a loss function that linearizes larger errors and squares smaller errors.

### Depth camera

In the experiment, using Kinect DK can make full use of its depth sensing and RGB image capture capabilities. Especially when dealing with transparent objects, Kinect DK's depth sensor can help reduce visual interference caused by transparency and reflection, thus improving the accuracy of object recognition. The detailed parameters are as follows.

Depth Sensor:

Sensor Type: Time-of-Flight (ToF).

PDF

Help

Resolution: 640 x 576 pixels Frame Rate: 5, 15, 30 FPS (configurable) Field of View (FOV): 75° x 65° (Horizontal x Vertical).

Depth Range: Indoor Mode: 0.5 m to 5.46 m

Outdoor Mode: 0.25 m to 2.88 m

RGB Camera:

Resolution: 3840 x 2160 pixels (4K) Frame Rate: 15, 30 FPS Field of View (FOV): 90° x 59° (Horizontal x Vertical) IMU (Inertial Measurement Unit): Gyroscope:  $\pm 2000$  °/s

Accelerometer:  $\pm 16$  g

Connectivity and Interface: USB 3.0 Type-C

Power Supply: 12V DC adapter

This table provides a comprehensive overview of the Kinect DK's capabilities, highlighting its depth sensing, RGB camera specifications, IMU parameters, and connectivity options.

## Color dynamic recognition technology (CDR)

Color dynamic recognition technology (CDR) generates 3 feature images by fusing multimodal datasets, corresponding to brightness, transparency and color saturation respectively. The size of each feature image is  $1 \times 100 \times 100$  (channel  $\times$  height  $\times$  width).

## Selective search algorithm

- 1) Each feature map (brightness, transparency, color saturation) is used as input, and the candidate region is generated by the Selective Search algorithm. Each feature map is processed separately to ensure the integrity of information under different modalities.
- 2) For each feature map, the Selective Search algorithm performs similarity calculation based on color similarity and region size similarity. The specific process is as follows:
  - 2.1 Initialize the candidate region set and equate the feature map to the initial candidate region.
  - 2.2 Calculate the color similarity metric and region size similarity metric.
  - 2.3 Select the region with the lowest similarity for segmentation and generate new candidate regions.

PDF

Help

- 3) Select candidate regions from each feature map and perform weighted fusion on these regions. The specific allocation method is:
  - 3.1 Select 200 candidate regions from the brightness feature map.  $00(r_1, r_2, \dots, r_2)$ .
  - 3.2 Select 150 candidate regions from the transparency feature map.  $50(r_{201}, r_{202}, \dots, r_3)$ .
  - 3.3 Select 150 candidate regions from the color saturation feature map.  $00(r_{351}, r_{352}, \dots, r_5)$ .
- 4) Weighted fusion of candidate regions extracted from each feature map.
 
$$\mathbf{Region}_{\text{combined}} = w_1 \cdot \mathbf{Region}_{\text{Lum}} + w_2 \cdot \mathbf{Region}_{\text{Par}} + w_3 \cdot \mathbf{Region}_{\text{Col}}.$$

$\mathbf{Region}_{\text{Lum}}$  represents the set of candidate regions of the brightness feature map.  $\mathbf{Region}_{\text{Par}}$  represents the set of candidate regions of the transparency feature map.  $\mathbf{Region}_{\text{Col}}$  represents the set of candidate regions of the color saturation feature map. The weights  $w_1, w_2$  and  $w_3$  are determined by the analytic hierarchy process (AHP) to reflect the relative importance of each modality.
- 5) The final 500 weighted fused candidate regions are used as the input of the R-CNN model for further feature extraction and target recognition.

## Quantification and statistical analysis

The quantification and statistical analysis for this study were primarily conducted using MATLAB R2022a and Python. The detailed steps and methods are as follows:

### Data Preprocessing

### Statistical Analysis

1. Descriptive Statistics: The mean, median, standard deviation, and variance of the normalized datasets were calculated using MATLAB's built-in functions.

```
mean_Lum = mean(Lum_norm);
median_Lum = median(Lum_norm);
std_Lum = std(Lum_norm);
var_Lum = var(Lum_norm);
mean_Par = mean(Par_norm);
```

PDF

Help

```
median_Par = median(Par_norm);  
  
std_Par = std(Par_norm);  
  
var_Par = var(Par_norm);  
  
mean_Col = mean(Col_norm);  
  
median_Col = median(Col_norm);  
  
std_Col = std(Col_norm);  
  
var_Col = var(Col_norm);
```

2. Correlation Analysis: Pearson correlation coefficients were calculated to determine the relationship between different features using the `corrcoef` function.

```
[R, P] = corrcoef([Lum_norm, Par_norm, Col_norm]);
```

3. Principal Component Analysis (PCA): PCA was performed to reduce the dimensionality of the data and identify the most significant features using the `pca` function.

```
[coeff, score, latent] = pca([Lum_norm, Par_norm, Col_norm]);
```

### Grasping Success Rate Analysis

1. Success Rate Calculation: The grasping success rate for different scenarios was calculated as the ratio of successful grasps to total attempts.

```
success_rate = sum(successful_grasps) / total_grasps;
```

2. Cycle Time Analysis: The average cycle time for grasping in different scenarios was calculated.

```
average_cycle_time = mean(cycle_times);
```

## Visualization

Plots and Graphs: Various plots were generated to visualize the data, including histograms, scatter plots, and PCA biplots.

```
figure;  
  
histogram(Lum_norm);  
  
title('Normalized Lum Values Histogram');  
  
figure;  
  
scatter3(Lum_norm, Par_norm, Col_norm);  
  
title('3D Scatter Plot of Normalized Features');  
  
figure;  
  
biplot(coeff(:, 1:2), 'Scores', score(:, 1:2));  
  
title('PCA Biplot');
```

## Acknowledgments

This work was funded by the Shandong Provincial Technology Innovation Guidance Program. Project number: [YDZX2023030](#).

## Author contributions

Conceptualization, C.G., methodology, Z.W. and C.C, formal analysis, C.G and J.S., writing–original draft, C.C, writing– review and editing, C.C and Z.W., and resources, Z.W., All authors have read and agreed to the final article.

## Declaration of interests

The authors declare no competing interests.

## Supplemental information

PDF

Help

 [Download: Download Acrobat PDF file \(105KB\)](#)

Document S1. TablesS1–S3.

 [Download: Download zip file \(10KB\)](#)

Data S1. Butterfly optimization algorithm, depalletizing and palletizing structure source code, related to Figure9.

#### Recommended articles

## References

- 1 R. Sadeghian, S. Shahin, S. Sareh  
**Vision-based self-adaptive gripping in a trimodal robotic sorting end-effector**  
IEEE Rob. Autom. Lett., 7 (2022), pp. 2124-2131, [10.1109/LRA.2022.3140793 ↗](#)  
[View in Scopus ↗](#) [Google Scholar ↗](#)
- 2 H. Zuo, Q. Li, H. Zheng, Y. Yang, X. Zhao  
**An Optically-Reconfigurable PUF Based on Logarithmic Photoreceptor of CMOS Dynamic Vision Sensors**  
IEEE Trans. Electron. Dev., 69 (2022), pp. 5395-5398, [10.1109/TED.2022.3191628 ↗](#)  
[View in Scopus ↗](#) [Google Scholar ↗](#)
- 3 Y. Liu, F. Zhou, W. Zhang, L. Yang, Z. Guo, H. Tan  
**Flexible Calibration Method for A Quad-directional Stereo Vision Sensor Based on Unconstraint 3D Target**  
IEEE Sensor. J., 4 (2023), pp. 1032-1038, [10.1109/JSEN.2023.3344594 ↗](#)  
[Google Scholar ↗](#)
- 4 Y. Li, Y. Chen, Y. Yang, Y. Wei  
**Passive particle jamming and its stiffening of soft robotic grippers**  
IEEE Trans. Robot., 33 (2017), pp. 446-455, [10.1109/TRO.2016.2636899 ↗](#)  
[Google Scholar ↗](#)
- 5 S. Phodapol, A. Harnkhamen, N. Asawalertsak, S.N. Gorb, P. Manoonpong

PDF

Help

## Insect tarsus-inspired compliant robotic gripper with soft adhesive pads for versatile and stable object grasping

IEEE Rob. Autom. Lett., 8 (2023), pp. 2486-2493, [10.1109/LRA.2023.3251186](https://doi.org/10.1109/LRA.2023.3251186) ↗

[View in Scopus](#) ↗ [Google Scholar](#) ↗

- 6 G. Gao, C.M. Chang, L. Gerez, M. Liarokapis  
**A pneumatically driven, disposable, soft robotic gripper equipped with multi-stage, retractable, telescopic fingers**

IEEE Trans. Med. Robot. Bionics, 3 (2021), pp. 573-582, [10.1109/TMRB.2021.3097143](https://doi.org/10.1109/TMRB.2021.3097143) ↗

[View in Scopus](#) ↗ [Google Scholar](#) ↗

- 7 Y. Yan, S. Guo, C. Lyu, D. Zhao, Z. Lin  
**Sea-based humanoid finger-functional parallel gripper with two actuators: Pg2 gripper**

IEEE Trans. Instrum. Meas., 72 (2022), p. 113, [10.1109/TIM.2022.3229695](https://doi.org/10.1109/TIM.2022.3229695) ↗

[View in Scopus](#) ↗ [Google Scholar](#) ↗

- 8 C. Nicholson-Smith, V. Mehrabi, S.F. Atashzar, R.V. Patel  
**A multi-functional lower-and upper-limb stroke rehabilitation robot**

IEEE Trans. Med. Robot. Bionics, 2 (2020), pp. 549-552, [10.1109/TMRB.2020.3034497](https://doi.org/10.1109/TMRB.2020.3034497) ↗

[View in Scopus](#) ↗ [Google Scholar](#) ↗

- 9 M. Mehdian, H. Rahnejat  
**A sensory gripper using tactile sensors for object recognition, orientation control, and stable manipulation**

IEEE Trans. Syst. Man Cybern., 19 (1989), pp. 1250-1261, [10.1109/21.44044](https://doi.org/10.1109/21.44044) ↗

[View in Scopus](#) ↗ [Google Scholar](#) ↗

- 10 O.A. Hay, M. Chehadeh, A. Ayyad, M. Wahbah, M.A. Humais, I. Boiko, L. Seneviratne, Y. Zweiri  
**Noise-tolerant identification and tuning approach using deep neural networks for visual servoing applications**

IEEE Trans. Robot., 39 (2023), pp. 2276-2288, [10.1109/TRO.2023.3235586](https://doi.org/10.1109/TRO.2023.3235586) ↗

[View in Scopus](#) ↗ [Google Scholar](#) ↗

- 11 Y. Wang, Y. Qiu, P. Cheng, J. Zhang  
**Hybrid CNN-transformer features for visual place recognition**

IEEE Trans. Circ. Syst. Video Technol., 33 (2022), pp. 1109-1122, [10.1109/TCSVT.2022.3212434](https://doi.org/10.1109/TCSVT.2022.3212434) ↗

[Google Scholar](#) ↗

- 12 R. Girshick, J. Donahue, T. Darrell, J. Malik

PDF

Help

## Rich feature hierarchies for accurate object detection and semantic segmentation

Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2014),  
[10.1109/CVPR.2014.81](https://doi.org/10.1109/CVPR.2014.81) ↗

[Google Scholar](#) ↗

13

S. Ghosh, P. Paral, A. Chatterjee, S. Munshi

## Histogram refined local ternary pattern-based bilateral LPP for vision sensor-based robot navigation guidance under challenging environments

IEEE Sens. Lett., 7 (2023), pp. 1-4, [10.1109/LSENS.2023.3272832](https://doi.org/10.1109/LSENS.2023.3272832) ↗

[Google Scholar](#) ↗

14

S. Tsuji, T. Kohama

## Proximity and contact sensor for human cooperative robot by combining time-of-flight and self-capacitance sensors

IEEE Sens. J., 20 (2020), pp. 5519-5526, [10.1109/JSEN.2020.2969653](https://doi.org/10.1109/JSEN.2020.2969653) ↗

[View in Scopus](#) ↗ [Google Scholar](#) ↗

15

H. Ohashi, T. Yasuda, T. Kawasetsu, K. Hosoda

## Soft Tactile Sensors Having Two Channels With Different Slopes for Contact Position and Pressure Estimation

IEEE Sens. Lett., 7 (2023), pp. 1-4, [10.1109/LSENS.2023.3268888](https://doi.org/10.1109/LSENS.2023.3268888) ↗

[Google Scholar](#) ↗

16

T. Weng, A. Pallankize, Y. Tang, O. Kroemer, D. Held

## Multi-modal transfer learning for grasping transparent and specular objects

IEEE Rob. Autom. Lett., 5 (2020), pp. 3791-3798, [10.1109/LRA.2020.2974686](https://doi.org/10.1109/LRA.2020.2974686) ↗

[Google Scholar](#) ↗

17

Z. Zhou, T. Pan, S. Wu, H. Chang, O.C. Jenkins

## Glassloc: plenoptic grasp pose detection in transparent clutter

2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE (2019),  
[10.1109/IROS40897.2019.8967685](https://doi.org/10.1109/IROS40897.2019.8967685) ↗

[Google Scholar](#) ↗

18

S. Li, X. Yin, C. Xia, L. Ye, X. Wang, B. Liang

## Tata: A universal jamming gripper with high-quality tactile perception and its application to underwater manipulation

2022 International Conference on Robotics and Automation (ICRA), IEEE (2022),  
[10.1109/ICRA46639.2022.9811806](https://doi.org/10.1109/ICRA46639.2022.9811806) ↗

PDF

Help



[Google Scholar ↗](#)

19

S. Li, H. Yu, W. Ding, H. Liu, L. Ye, C. Xia, X. Wang, X.P. Zhang

## Visual–Tactile Fusion for Transparent Object Grasping in Complex Backgrounds

IEEE Trans. Robot., 3 (2023), pp. 7568-7574, [10.1109/TRO.2023.3286071 ↗](#)

[View in Scopus ↗](#) [Google Scholar ↗](#)

20

J. Jiang, G. Cao, T.T. Do, S. Luo

## A4T: Hierarchical affordance detection for transparent objects depth reconstruction and manipulation

IEEE Rob. Autom. Lett., 7 (2022), pp. 9826-9833, [10.1109/LRA.2022.3191231 ↗](#)

[View in Scopus ↗](#) [Google Scholar ↗](#)

21

Y. Ji, L. Wang, W. Wu, H. Shao, Y. Feng

## A method for LSTM-based trajectory modeling and abnormal trajectory detection

IEEE Access, 8 (2020), pp. 104063-104073, [10.1109/ACCESS.2020.2997967 ↗](#)

[View in Scopus ↗](#) [Google Scholar ↗](#)

22

J. Ichnowski, Y. Avigal, J. Kerr, K. Goldberg

## Dex-nerf: Using a neural radiance field to grasp transparent objects

Preprint at

arxiv (2021), [10.48550/arXiv.2110.14217 ↗](#)

[Google Scholar ↗](#)

23

T. Sun, G. Zhang, W. Yang, J.-H. Xue, G. Wang

## TROSD: A New RGB-D Dataset for Transparent and Reflective Object Segmentation in Practice

IEEE Trans. Circ. Syst. Video Technol., 33 (2023), pp. 5721-5733, [10.1109/TCSVT.2023.3254665 ↗](#)

[View in Scopus ↗](#) [Google Scholar ↗](#)

24

M. Sridharan, P. Stone

## Structure-based color learning on a mobile robot under changing illumination

Aut. Robots, 23 (2007), pp. 161-182, [10.1007/s10514-007-9038-7 ↗](#)

[View in Scopus ↗](#) [Google Scholar ↗](#)

25

S. Yang, W. Zhang, R. Song, W. Lu, H. Wang, Y. Li

PDF

Help

# Explicit-to-Implicit Robot Imitation Learning by Exploring Visual Content Change

IEEE/ASME Trans. Mechatron., 27 (2022), pp. 4920-4931, [10.1109/TMECH.2022.3166993](https://doi.org/10.1109/TMECH.2022.3166993) ↗

[View in Scopus](#) ↗ [Google Scholar](#) ↗

26 T.T. Nguyen, L. Gordon-Brown

## Constrained fuzzy hierarchical analysis for portfolio selection under higher moments

IEEE Trans. Fuzzy Syst., 20 (2011), pp. 666-682, [10.1109/TFUZZ.2011.2181520](https://doi.org/10.1109/TFUZZ.2011.2181520) ↗

[Google Scholar](#) ↗

27 M.-T. Le, J.-J.J. Lien

## Lightweight Robotic Grasping Model Based on Template Matching and Depth Image

IEEE Embed. Syst. Lett., 14 (2022), pp. 199-202, [10.1109/LES.2022.3181892](https://doi.org/10.1109/LES.2022.3181892) ↗

[View in Scopus](#) ↗ [Google Scholar](#) ↗

28 J. Tang

## Research on Visual Localization and Gripping Technology of Robotic Arm Based on Deep Learning

Int. J. Comput. Sci. Inf. Technol., 2 (2024), pp. 504-511, [10.62051/ijcsit.v2n1.53](https://doi.org/10.62051/ijcsit.v2n1.53) ↗

[Google Scholar](#) ↗

### Cited by (0)

2 These authors contributed equally

3 Lead contact

© 2024 The Author(s). Published by Elsevier Inc.

PDF

Help



All content on this site: Copyright © 2024 Elsevier B.V., its licensors, and contributors. All rights are reserved, including those for text and data mining, AI training, and similar technologies. For all open access content, the Creative Commons licensing terms apply.



PDF

Help