Fast and Comfortable Interactive Robot-to-Human Object Handover

Chongxi Meng[1,2], Tianwei Zhang[2], Tin Lun Lam[1,2,†]

***Abstract*— Transferring tools and objects to human hands is an important ability of collaborative robots. Most of the existing approaches focus on handover affordance, however, the comfort of receiving objects with human hands is often neglected. In this paper, we use advanced deep learning models to pre-generate handover target configurations that are convenient for human grasping based on the characteristics of the objects and tools, and then the robot grasps and passes the objects to the human. Experimental results on a mobile collaborative robot show that our proposed framework can robustly and efficiently deliver different shapes and types of objects to a human hand of any pose within the robot's field of view in a target pose that is convenient for grasping and can quickly deliver objects to a new target location even after the human hand moves to a new position.**

I. Introduction

It is a significant application that transfers objects between robots and people for human and robot cooperation. Robots can improve factory efficiency by handing tools to workers, or they can hand household items to people who are inconvenient to move. It seems natural for people to pass the object to the receiver quickly, accurately, and as effortlessly as possible. However, many challenges remain to achieve a fluent robot-to-human object handover for robots.

Firstly, the robot needs to consider how to grasp the object suitably that humans will receive. Because the grasp of a robot influences the grasp of the human. Humans can only grasp the object on the unrestricted portion of the object. Special attention should be paid to objects of particular use and objects with tips, such as water cups and scissors. Robots must ensure that people can safely complete the handover. Secondly, unlike placing a static object on a plane, the process of handover may involve hand movement, which may result in unsuccessful handover attempts. Therefore, the robot needs to track the human hand in real-time to ensure that the robot arm can complete the handover after the human hand moves. it can ensure that the robot arm does not collide with the human hand during the movement process. Finally, after the end effector moves to the target point, the robot needs to detect whether the object is successfully grasped by the receiver. Because the robot has to select whether to release the gripper and return to the initial position after the human hand retracts. Usually, the robot uses the data obtained by the end force sensor to judge whether the object is successfully grasped by the receiver.

Fig. 1: Robots transport object that human hands can grasp comfortably. Our system first predicts a suitable grasp for the human hand, then based on the human hand tracking and reconstruction, it delivers the object to the human and can adapt to the scene of the human hand moving.

In response to the above issues, we propose a real-time interactive robot-to-human handover framework. The contributions are as follows:

- A vision-based robot-to-human handover system, the robots can track human hands in real-time and quickly deliver objects to the receiver.
- Suitable handover posture, the robot generates both a safe human hand grasp posture and a robot grasp posture on the object.
- Robust object handover strategy, The robot uses a multi-threaded, reactive strategy to complete the handover of different objects with a high success rate.

II. Related Works

A recent survey summarizes the progress of research on robot handover capabilities [1]. It can be seen that the results of human-robot handover research are rising year by year. Handover is usually divided into two cases, robot-to-human and human-to-robot. This paper restricts the research scope to the robot-to-human handover. During the handover process, humans should grasp the appropriate, safe part of the object. Functional interaction of object parts with humans is usually judged based on the object's affordance. Here affordance means the intuitive usage of the object [2]. Humans reason about the affordance of objects mainly through vision,

This work was supported by the National Natural Science Foundation of China (62073274), and the funding AC01202101103 from the Shenzhen Institute of Artificial Intelligence and Robotics for Society

[1]School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen.

[2]The Shenzhen Institute of Artificial Intelligence and Robotics for Society (AIRS).

[†]Corresponding author Tin Lun Lam tllam@cuhk.edu.cn

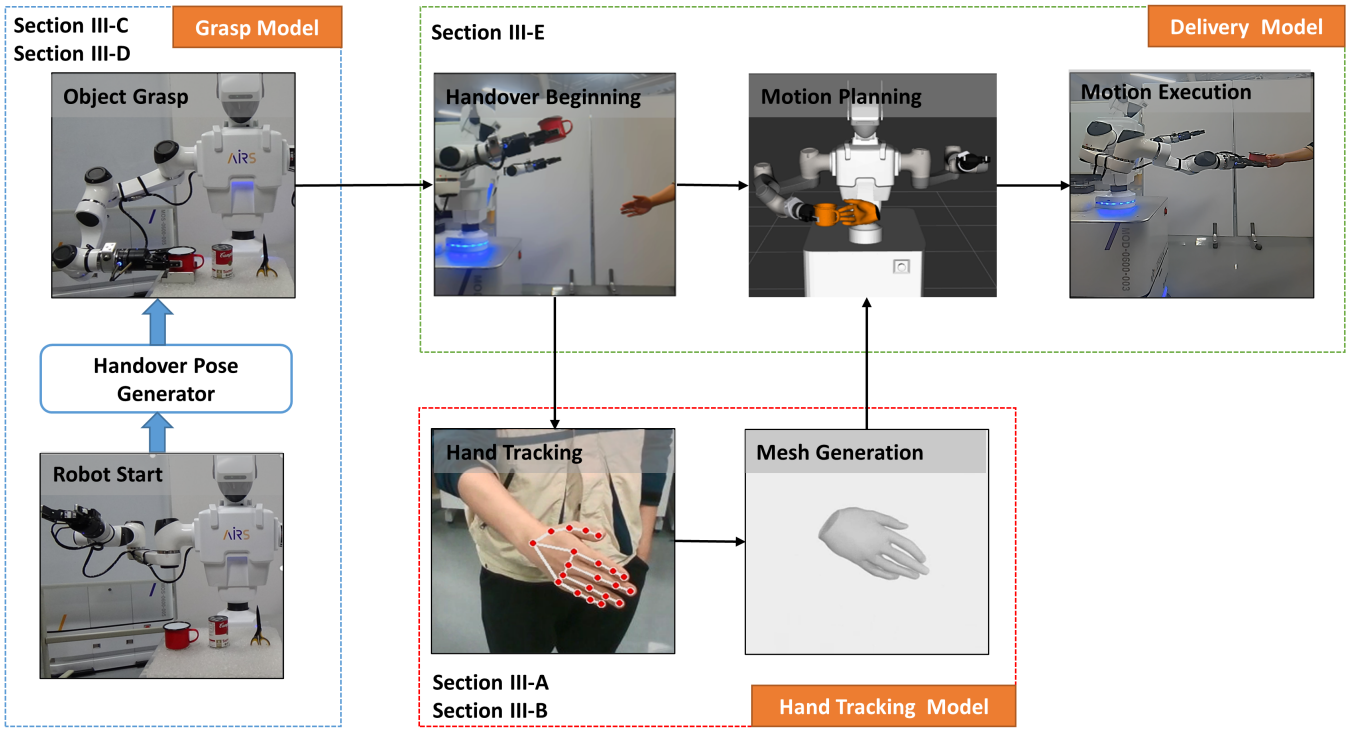

Fig. 2: An overview of the proposed handover system. The system is divided into three modules namely grasp, hand tracking, and delivering. The robot first predicts the grasping posture of objects commonly used by the human hand. Based on this prior, it executes a feasible grasp and turns the waist. When it tracks hand and generates mesh successfully, the robot calculates the ideal handover points and delivers the object based on a reactive strategy.

so there is a lot of work based on RGB or point clouds [3][4]. Besides Ardon *et al.* uses Markov logic networks to learn semantic relationships between features such as properties, location, and grasping functions of objects [5]. while later using CNN to extract features of the inferred object from RGB, querying specific grasps from the trained knowledge graph. Corona *et al.* uses a single RGB image with one or more objects to predict how humans will naturally grasp these objects [6]. However, these works do not consider the constraint of robotic grasping. Our system is close to the work of Ardon *et al.* [7]. But the difference is that the pose of the human hand can be changed arbitrarily in our scenario.

The handover must take place in a location that is accessible to both agents. In robot-to-human handover tasks, many works assume that the human hand does not move during the process, so the handover point does not change. In reality, however, the human hand usually moves. Therefore, we propose a method to determine the handover point based on the estimation of human hand pose. To determine the human hand position in space, the robot needs real-time human hand tracking. MediaPipe [8] firstly determines the position of the hand using the hand detector. Then it inputs the bounding box containing the hand into the landmark model and generates the 2.5 dimension hand coordinate frame. The human hand segmentation [9] in human-to-robot handover is similar to hand tracking in robot-to-human handover. The first crop is point clouds containing only hands and objects, and then predicts a hand mask based on the RGB image. They both use the feature pyramid network as the backbone, and the network outputs a per-pixel segmentation. Each point in the point clouds can be labeled as a hand or object using a hand mask.

To improve system safety and avoid any contact between humans and robots, Rosenberger *et al.* add a body segmentation module based on refinement [10] [11]. Zhang *et al.* proposed to use human pose estimation method to find dynamic human objects when robot is moving [12] [13]. Zhou *et al.* gives a real-time 3D hand reconstruction method for the MANO model [14] [15], and the method is divided into two modules. The DetNet is used to regress the 3D hand joint positions. the IKNet solves the pose and shape parameters in the forward channel and inputs the MANO model for human hand reconstruction. Hasson *et al.* reconstructs hands and objects simultaneously from RGB images [16]. The authors give a Pytorch version of the differentiable MANO model and create a dataset with human-manipulated object actions.

III. System Framework And Methods

In this study, we propose a fully automatic system to complete the handover from robot to human. The method pipeline is shown in Fig. 2. After finding people, the robot generates the grasping posture for the handover task and executes it. In addition, the robot detects the human hand and calculates the best handover point. Finally, the robot plans and executes the trajectory through the motion planner. When a person has grasped the object, the robot actively releases the object. The next arrangement is as follows.

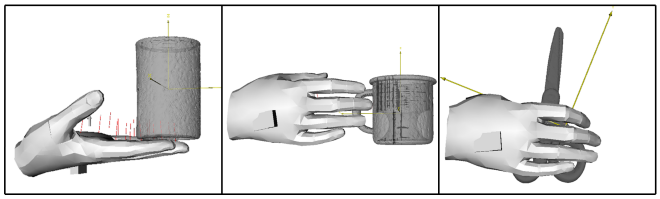

Fig. 3: Examples of a suitable grasp type of hand mesh generated on an object.

1) Quick and accurate calculation of the handover point. Through real-time hand pose estimation, the robot can output the pose of the target when the hand is moving.

2) Optimal grassing and delivery, the robot not only makes people grasp objects comfortably but also comfortably delivers objects.

3) By constructing a hierarchical and reactive execution strategy, we can solve the uncertainty and change of the current human state.

A. *Hand detection and tracking*

Our visual input comes from an RGB-D camera, and the image pair is denoted as $f_i = (\mathcal{C}_i, \mathcal{D}_i)$, where $\mathcal{C}_i : \Omega \to \mathbb{R}^3$ and $\mathcal{D}_i : \Omega \to \mathbb{R}$ stand for the i-th color image and depth image respectively. $\Omega \subset \mathbb{R}^2$ is the image domain. So we use $\mathcal{C}_i$ to predict hand skeletons. Each human hand is detected by a learning-based algorithm, and hand detection is refined based on geometric relationships. MediaPipe hand is a high-fidelity hand and finger tracking solution, and it uses machine learning to infer hands joint point information. Through this method, we obtain 2D hand with joint point pixels$\{(u_1, v_1), (u_2, v_2), ..., (u_{21}, v_{21})\}$ in real time. The numerical sequence number of joints is the same as this article [17]. The next step is to construct the coordinate transformation of the end effector coordinate system$\mathcal{F}_{rf}$, camera coordinate system $\mathcal{F}_{rc}$ and hand coordinate system $\mathcal{F}_{rh}$ in the robot coordinate system $\mathcal{F}_r$.

We store MediaPipe outputs containing joint points P with confidence $c > 0.5$ as a set $H_j = \{{}_1P^j, {}_2P^j, ..., {}_{21}P^j\}$, where $j \in \{0, 1\}$ is the flag denoting which side of the hand was detected in $\mathcal{C}_i$. The 3D point P relative to $\mathcal{F}_r$ is computed by the pinhole camera model:

$$ {}_kP_c = (\frac{u_k - c_x}{f_x} d_k, \frac{v_k - c_y}{f_y} d_k, d_k)^T \tag{1} $$

where f_x, f_y, c_x, c_y are the intrinsic parameters of the camera, and ${}_kP_c, u_k, v_k$ are pixel value corresponding to the kth point. Based on transformation tree in robot, we get $\mathcal{F}_{rc}$ to calculate P_r. Our work contributes to a comfortable handover for humans. The robot needs a 6D target as motion planning inputs, so we need to define $\mathcal{F}_{rh}$ to describe arbitrary hand pose. According to the article [18], index finger root, ring finger root, and wrist have little deformation when moving around. So we select points ${}_0P_r$, ${}_5P_r$, ${}_{13}P_r$ to build hand frame $\mathcal{F}_{rh} = [\vec{x}, \vec{y}, \vec{z}, \vec{p}]$ and specify the inner side of the palm as positive direction of z-axis. The x-axis is defined as the connecting line between ${}_0P_r$ and ${}_5P_r$, and the y-axis is defined according to the right-hand rule.

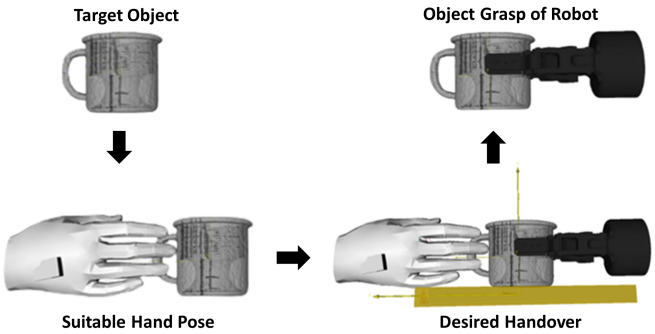

Fig. 4: Given a target object, we reconstruct the joints of the hand manipulating object. Based on the constrained object, graspit creates a safe and desired gripper pose. Finally, the robot solves the global pose through coordinate transformation and executes the planned trajectory.

$$ \vec{z} = (\frac{({}_5P_r - {}_0P_r) \times ({}_{13}P_r - {}_0P_r)}{\|({}_5P_r - {}_0P_r) \times ({}_{13}P_r - {}_0P_r)\|})^T \tag{2} $$

B. *Mesh Generation*

We choose MANO [15] as the hand model to be driven by the output of Section III-A, because model-based hand reconstruction is significantly better than the model-free. The surface mesh of MANO can be fully deformed and posed by standard linear blend skinning function as

$$ \mathcal{M}(\theta, \beta) = \mathcal{W}(\mathcal{T}(\beta, \theta), \mathcal{J}(\beta), \theta, \omega) \tag{3} $$

where $\mathcal{M}(\theta, \beta) \in \mathbb{R}^{773\times3}$ is hand mesh surface, $\beta \in \mathbb{R}^{10}$ and $\theta \in \mathbb{R}^{16\times3}$ are shape parameters and pose parameters, ω is a constant. So our work is to use H_j to estimate β and θ, called hand inverse kinematics.

Manolayer [16] is a differentiable Pytorch layer that maps directly from β and θ to $\mathcal{M}$ and joint positions, which means we can use an automatic differentiation engine to optimize β and θ. Therefore, we first initialize β and θ to 0, then obtain the 3D coordinates of the hand joints through the forward channel of Manolayer. We regard $\mathcal{F}_{rh}$ and H_j as a truth value, then calculate the loss between truth and output of Manolayer. When the first frame is initialized successfully, β and θ are used as the initial value of the next optimization. We reversely update the β and θ to generate the current hand mesh.

C. *Desired Hand Pose*

Formally, given a CAD model T, we need a model $\mathcal{N}$ that provides a hand pose P and shape V mentioned in equation (3), and grasp type C for an object in T:

$$ \mathcal{N} : T \Longrightarrow \{C, V, P\} \tag{4} $$

We choose Ganhand [6] as the hand generation method, and there are three reasons. First, Ganhand predicts the optimal human hand grasp type and generates a hand mesh on the object. Second, Ganhand and our work both use the YCB object set so that We could easily reduce the generalization error. Third, the author uses the real grasped objects by

Algorithm 1 Reactive strategy

Input: Hand Pose $\mathcal{F}_{rh}$ Object Pose $\mathcal{F}_{rb}$ Effector Pose $\mathcal{F}_{rt}$ Robot Pose $\mathcal{F}_r$

$flag1 \leftarrow 1, \quad flag2 \leftarrow 0, \quad flag3 \leftarrow 0$
for $i \in$ all objects **do**
  if $flag1 = 1$ **then**
    $\mathcal{F}_{rb}^{(i)} \leftarrow object\_grasp(\mathcal{F}_r)$
    $\mathcal{F}_r \leftarrow turn\_waist(\mathcal{F}_r)$
    $flag1 \leftarrow 0, \quad flag2 \leftarrow 1$
  end if
Thread1:
  if $hand\_motion(\mathcal{F}_{rh}) = True$ **then**
    $\mathcal{F}_{rt} \leftarrow robot\_stop(\mathcal{F}_r)$
    Thread2 **Stop**
    $flag2 \leftarrow 1, \quad flag3 \leftarrow 0$
  end if
Thread2:
  if $flag2 = 1$ **then**
    $\mathcal{F}_{rb}^{(i)} \leftarrow plan\_execution(\mathcal{F}_{rh})$
    $flag2 \leftarrow 0, \quad flag3 \leftarrow 1$
  end if
  if $flag3 = 1$ **then**
    $\mathcal{F}_{rt} \leftarrow open\_gripper(\mathcal{F}_r)$
    $flag3 \leftarrow 0, \quad flag1 \leftarrow 1$
    Thread1 **Stop**
    $\mathcal{F}_r \leftarrow turn\_waist(\mathcal{F}_r)$
  end if
end for
return $\emptyset$

human hands to train the model. Unlike their work, we assume that the global pose of the object is known and the object lies flat on the table. Because our task only needs to consider the relative relationship between the human hand and the object. We select images that meet the assumptions to predict hand grasp types. Fig. 3 shows examples of hand grasp configurations.

D. *Handover Pose and Object Grasp*

Based on the desired pose of the hand, the robot plans the pose of the gripper and grasps the object. the object being grasped is of the same size and shape as the set of YCB objects. We firstly set the geometric center of the objects as the coordinate origin and the table as the object placement plane. We import the C, V, P, T into the graspit. The graspit simulator includes fast collision detection system that can generate grasp pose based on the gripper model. To achieve better presentation, we prefer to choose the grasping posture opposite to hands in addition to preventing collision with hands. One reason is that the caliber of our gripper is small, and the chance of grasping failure from other postures will be greatly improved.

The coordinate frame $\mathcal{F}_{rb}$ of the object is known, and the target transformation $\mathcal{F}_{bt}$ between the end effector and object is calculated by model $\mathcal{N}$. So the grasping task is defined as that robot plans from initial pose $\mathcal{F}_{bi}$ to pose $\mathcal{F}_{rb}\mathcal{F}_{bt}$. Fig. 4 shows examples of generating robot gripper configurations.

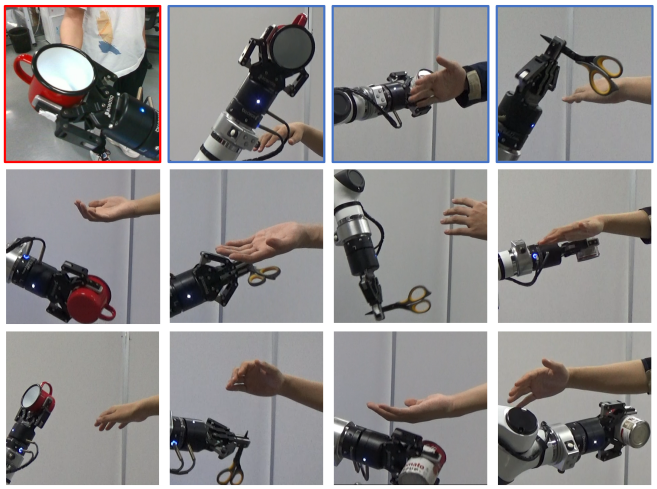

Fig. 5: Handover fail examples in our experiment. The failure case with a red border means that the camera is blocked by an object and can not find a human hand in the new position. The blue border means the human hand and robotic arm move at the same time and occlude the camera. Other means the path is too long to deliver the object within the allotted time.

E. *Reactive strategy and execution*

Because human behavior is uncertain, the robot needs to decide what to do next according to the current state of humans and itself. We design a task model presented in the Algorithm 1 based on reaction strategy, and the **Thread1** and **Thread2** are respectively responsible for detecting whether the hand is moving and executing robot motion.

Firstly, we divide the whole process of handover into three states: grasp, deliver and release. At runtime, we repeatedly check the execution conditions of each state to determine which state to enter. If the robot gripper opens, the robot enters the grasp state. If the robot completes grasp, the robot enters the delivered state. If the robot arm reaches the desired position, the robot enters the release state. For safety and to solve the problem that objects cannot be accurately delivered to the human hand after the change of the human hand in the previous work, the robot needs to determine whether the human hand is moving in real time. We use the hexahedral volume formed by three points ${}_0P_r$, ${}_5P_r$, ${}_{13}P_r$ to form a plane for a small period as a way to evaluate the movement of the human hand. The execution of the robot motion planning is done through an inverse kinematics solver and self-collision detection.

IV. Experiments Results and Evaluations

In this work, we used the Moying dual-arm robot. Both arms of this robot have 6 degrees of freedom, a maximum range of motion of 705 mm, and a maximum hand load of 3 kg. The end effector of the robot arm is equipped with a two-finger parallel gripper. The gripper can pick up objects with a maximum width of 8 cm. A Robotiq FT85 torque sensor can be added to the end. The head of the robot is installed with Intel RealSense D435 RGB-D camera. The sensors on

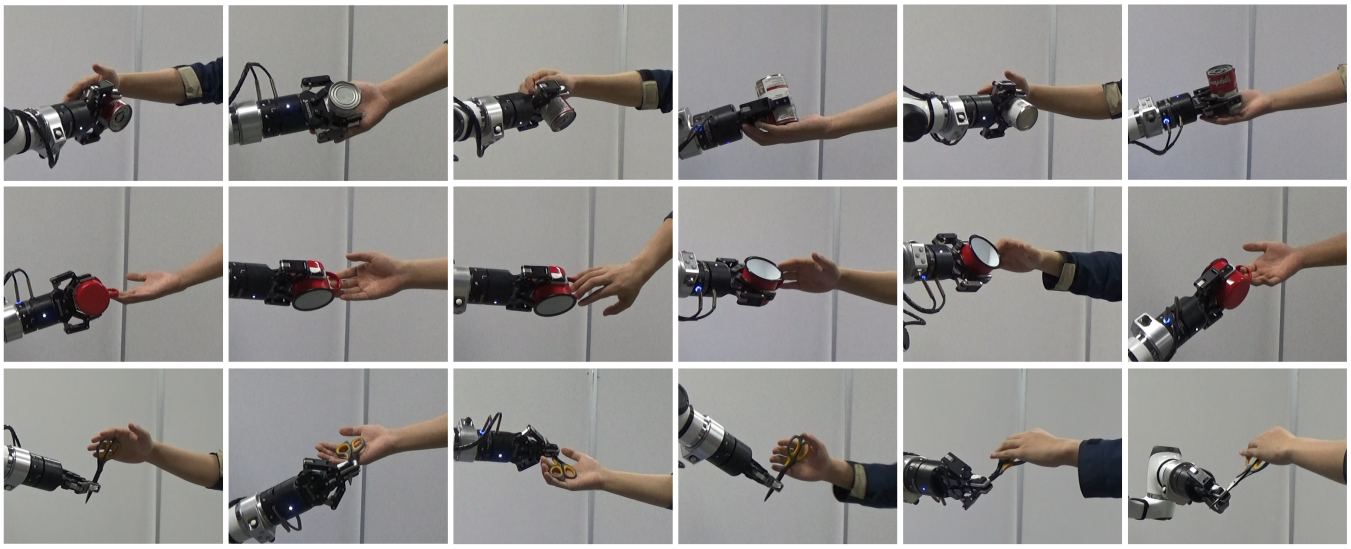

Fig. 6: Handover success example. Three objects of different sizes and shapes were used: a can, a mug, and scissors. The same user performed all experiments. For any hand pose, the robot successfully delivered the object to the human.

the robot have been calibrated. The motion planning and execution module runs on the robot body computer. The hand detection and mesh generation module run on a desktop computer with Intel Core^{TM} i9-9980XE CPU @ 3.00 GHz $\times$ 36, 128 GB System memory, and one GeForce RTX 2080 Ti GPU. The communication between the two computers is done through ROS. Transformations Frames Tools in ROS update $\mathcal{F}_{rh}$, $\mathcal{F}_{rb}$, $\mathcal{F}_{rt}$, $\mathcal{F}_{r}$.

A. *Handover Experiments*

We conducted experiments to test the performance of our system in response to different types of handover with arbitrary hand poses. Specifically, we looked at three different objects. But, if the object model is known, our method can be applied to any graspable object. As shown in Fig. 6, the objects were a mug, scissors, and a can from the YCB object [19] that fit the Moying robot gripper. The three objects are placed on the operating table behind the waist of the robot respectively, and the initial pose is known.

For each object, we ensure that the human hand is within the workspace of the robot. The pose of the human hand can be arbitrary, and the robot needs to send the object to the human hand. When the position of the human hand changes, the robot needs to respond in time, adjust the posture of the end effector and deliver the object to the human again. We conducted 30 experiments on each object. Each experiment is divided into three stages. The first stage is that the end effector sends the object to the human hand from the initial position. The second stage is that after the first movement of the hand, the manipulator will redeliver the object to the hand from the current position. The third stage is the same as the second stage. The purpose of this process is to test whether our reactive strategy and perception system can cope with the problem of hand movement.

To ensure the fluency of people's subjective experience, we need to set the maximum time for one delivery, if it exceeds this time, we judge the experiment to fail. We refer to the average time taken for a delivery in previous work [9] to set the metric. Our metric is that a handover must be completed within 8 seconds and free collisions with the human hand. Some of our successful examples in the experiment are shown in Fig. 6. It can be seen that we realize handover based on arbitrary hand pose.

TABLE I: Success rate of 30 experiments per object

Object	mug	can	scissors
Hand Still	1.0	0.96	1.0
First Hand Move	0.90	0.86	0.96
Second Hand Move	0.76	0.76	0.86

B. *Evaluations*

For three object handover, the robot execution success rate and execution time are shown in the table I. As can be seen from the data distribution in the table, the success rate of our system does not depend on the handover object. There are two reasons for the 20.7% failure rate after two hand position changes First, the planning algorithm we use is not globally optimal. Because the globally optimal planner will consume too much time, which will affect people's subjective feelings. Exceeding the specified time in our experiments will be considered a task failure. At the same time, due to the prevention of collision with the human hand, we cannot just use the inverse kinematics algorithm to solve the desired pose. So we choose RRT-connect to reduce planning time. However, this algorithm does not guarantee the globally optimal path, which means that the path will become longer, which will also lead to timeouts. Second, after the human hand moves, some parts of the robot arm may occlude the human hand in the new position, causing the robot to fail to correctly estimate the pose of the human hand. Besides,

when the robotic arm delivers the object, it just blocks the moving hand and cannot stop according to the reactive strategy. However, our system does not cause harm to people in these failure cases, such as collision with a human body or hand. Our next work will also focus on solving the problems in these three aspects to achieve a safer, faster, and more comfortable robot-to-human handover.

V. Conclusion

In this paper, we have presented a robot-to-human handover system for handover objects to humans in arbitrary poses quickly and comfortably. When the hand moves, our system can still accurately send objects to the hand and limit the single execution time to less than 8s. Even after the hands are moved twice, the interactive handover can still be realized with a high success rate. At the same time, our system eliminates areas that are not suitable for people to grasp, so that people can easily get the required objects. This work can help people with limited mobility to take objects or help factory workers improve efficiency. In the future, we will focus on optimizing the trajectory of the end effector in the process of handover based on considering the human hand model and applying our work to multi-robot systems[20].

References

[1] V. Ortenzi, A. Cosgun, T. Pardi, W. P. Chan, E. Croft, and D. Kulić, "Object handovers: a review for robotics," *IEEE Transactions on Robotics*, 2021.

[2] Y. Zhu, A. Fathi, and L. Fei-Fei, "Reasoning about object affordances in a knowledge base representation," in *European conference on computer vision*. Springer, 2014, pp. 408–424.

[3] T.-T. Do, A. Nguyen, and I. Reid, "Affordancenet: An end-to-end deep learning approach for object affordance detection," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 5882–5889.

[4] S. Deng, X. Xu, C. Wu, K. Chen, and K. Jia, "3d affordancenet: A benchmark for visual object affordance understanding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1778–1787.

[5] P. Ardón, È. Pairet, R. P. Petrick, S. Ramamoorthy, and K. S. Lohan, "Learning grasp affordance reasoning through semantic relations," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4571–4578, 2019.

[6] E. Corona, A. Pumarola, G. Alenya, F. Moreno-Noguer, and G. Rogez, "Ganhand: Predicting human grasp affordances in multi-object scenes," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 5031–5041.

[7] P. Ardón, M. E. Cabrera, E. Pairet, R. P. Petrick, S. Ramamoorthy, K. S. Lohan, and M. Cakmak, "Affordance-aware handovers with human arm mobility constraints," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3136–3143, 2021.

[8] F. Zhang, V. Bazarevsky, A. Vakunov, A. Tkachenka, G. Sung, C.-L. Chang, and M. Grundmann, "Mediapipe hands: On-device real-time hand tracking," *arXiv preprint arXiv:2006.10214*, 2020.

[9] W. Yang, C. Paxton, A. Mousavian, Y.-W. Chao, M. Cakmak, and D. Fox, "Reactive human-to-robot handovers of arbitrary objects," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 3118–3124.

[10] P. Rosenberger, A. Cosgun, R. Newbury, J. Kwan, V. Ortenzi, P. Corke, and M. Grafinger, "Object-independent human-to-robot handovers using real time robotic vision," *IEEE Robotics and Automation Letters*, vol. 6, no. 1, pp. 17–23, 2020.

[11] G. Lin, A. Milan, C. Shen, and I. Reid, "Refinenet: Multi-path refinement networks for high-resolution semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1925–1934.

[12] T. Zhang and Y. Nakamura, "Posefusion: Dense rgb-d slam in dynamic human environments," in *International Symposium on Experimental Robotics*. Springer, 2018, pp. 772–780.

[13] T. Zhang, H. Zhang, Y. Li, Y. Nakamura, and L. Zhang, "Flowfusion: Dynamic dense rgb-d slam based on optical flow," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 7322–7328.

[14] Y. Zhou, M. Habermann, W. Xu, I. Habibie, C. Theobalt, and F. Xu, "Monocular real-time hand shape and motion capture using multi-modal data," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 5346–5355.

[15] J. Romero, D. Tzionas, and M. J. Black, "Embodied hands: Modeling and capturing hands and bodies together," *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, vol. 36, no. 6, Nov. 2017.

[16] Y. Hasson, G. Varol, D. Tzionas, I. Kalevatykh, M. J. Black, I. Laptev, and C. Schmid, "Learning joint reconstruction of hands and manipulated objects," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 11 807–11 816.

[17] M. Zhang, Z. Zhou, and M. Deng, "Cascaded hierarchical cnn for 2d hand pose estimation from a single color image," *Multimedia Tools and Applications*, pp. 1–19, 2022.

[18] T. Kurihara and N. Miyata, "Modeling deformable human hands from medical images," in *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 2004, pp. 355–363.

[19] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, "The ycb object and model set: Towards common benchmarks for manipulation research," in *2015 international conference on advanced robotics (ICAR)*. IEEE, 2015, pp. 510–517.

[20] X. Zhang, L. Yan, T. L. Lam, and S. Vijayakumar, "Task-space decomposed motion planning framework for multi-robot loco-manipulation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 8158–8164.