Robot navigation and grasping for household scenarios: a point cloud-centric approach

Shixing Wan, Zhendong Chen, Shuai Zhao*
Department of Computer Science and Engineering, Sun Yat-Sen University, Guangzhou 510275, Guangdong, China

ABSTRACT

The emerging application of robots in household scenarios provides a promising solution for daily housekeeping, warehousing and healthcare. In such scenarios, the robot needs capabilities for environment perception, object searching, navigation, and object manipulation. However, existing solutions often focus on a single functionality and cannot handle complex and unseen environments, imposing a huge barrier to collaboratively processing information in complex environments. In this paper, we propose a holistic robot control framework that allows a robot to percept, search, navigate, and grasp target objects at specified locations, providing a complete solution with the major functionalities for smart home robots. To tackle the issue of complex background interference in the environment, we developed a perception module that combines instance segmentation with point cloud clustering to extract spatial information of the target object. Based on the spatial information obtained by the perception module, we constructed an object-searching algorithm that uses related objects as clues that accelerate the object-searching process in unseen environments. In addition, a new grasp detection model is presented that explicitly considers the information of both the objects and backgrounds to reduce the interference of background on grasp detection, improving the grasp success ratio. Results from simulation experiments demonstrate that the proposed approach significantly outperforms the baseline method in terms of both efficiency and success rate.

Keywords: Robot navigation, grasp detection, environment perception, smart housekeeping

1. INTRODUCTION

Navigation and grasping are vital aspects of robotics technology, with real-world applications spanning diverse sectors such as warehousing, healthcare, and smart home management. Currently, such technology is predominantly applied in laboratory and restricted industrial settings. To accelerate the deployment of service robots in real-world scenarios, the Mobile Robot Grasping and Navigation Challenge 2023 (MRGNC2023)[1] provides a challenge to evaluate the perception, navigation, and grasp capacity of robots in household scenarios. This challenge aims for robots to locate specified objects in unseen household scenarios, grasp them, and complete subsequent tasks, such as placing them in a designated location. For target-driven navigation tasks, efficiently exploring an unseen environment and accurately extracting target information from complex backgrounds present significant challenges. In complex environments, the relative positioning between robots and objects cannot be predetermined, and the objects vary in shape, size, and orientation. This diversity necessitates that grasp detection methods possess strong generalizability and robustness.

To tackle this challenge, we propose a framework namely Point Cloud-Centric Robotic Grasping and Navigation (PCRGN), which consists of the following three components:

Perception. For target-driven tasks, it is crucial to extract the information of target from the environment to minimize background interference. Instance segmentation technology can separate each object from its surroundings, but image-based segmentation often struggles with accurate boundary delineation, shown as in Section 4.1. Background noise can significantly impact both localization and grasp detection. Therefore, we have designed a perception module that uses RGB-D as input, where image information is utilized for identification and spatial information for noise reduction, allowing for more accurate extraction of object details.

Search. When our perception system is capable of detecting objects and locating them within the field of view, the perception model can provide accurate information about the position of the object. With the availability of maps, path-planning algorithms to reach given targets are well-developed[2-4]. However, exploring and discovering the target remains a

*zhaosh56@mail.sysu.edu.cn

International Conference on Optics, Electronics, and Communication Engineering (OECE 2024), edited by Yang Yue, Proc. of SPIE Vol. 13395, 133950X · © 2024 SPIE · 0277-786X · Published under a Creative Commons Attribution CC-BY 3.0 License · doi: 10.1117/12.3049707

challenging issue that needs to be addressed[5]. In typical scenarios, objects have consistent placements, such as on tables, chairs, or nightstands. By leveraging spatial relations among objects, potential search areas can be prioritized, thereby optimizing the search process. Based on this we design our search module by cule as related objects.

Grasp. Upon reaching the vicinity of an object, the robot can view it from various angles, resulting in a background that is not as fixed as in typical grasping tasks. The perception module provides the capability to isolate and extract information about the object. This object information is then used to predict grasping actions, while background information is utilized for collision detection. Additionally, to further enhance the capabilities of our grasping model, we have designed an automated process for collecting and annotating datasets.

Our framework enables robots to perform navigation and grasping tasks in complex and unseen household environments. It allows for control over different modules based on the state of the agent and utilizes point cloud data to facilitate interaction between these modules. Experimental results demonstrated our approach surpassing baseline methods by a significant 160% margin, securing first place in the competition. For details on the specific case, please visit at https://github.com/omitted001/RobotN-G.

2. RELATED WORK

2.1 Perception via instance segmentation

Image segmentation is a fundamental problem in computer vision, it involves partitioning images into multiple segments and objects[6]. Instance segmentation is a technique that identifies and segmentation each specific instance of objects within an image. He et al.[7] proposed Mask R-CNN, a two-stage instance segmentation model that first generates Regions of Interest (ROIs) and subsequently classifies and segments these ROIs. While Mask R-CNN effectively classifies and segments each instance in an image, its two-stage process does not achieve real-time speeds[8]. Bolya et al.[8] proposed YOLACT, the first real-time instance segmentation model, that adopts a parallel structure, lightweight assembly process, and a novel Fast NMS (Non-maximum suppression) approach to reduce the time overhead of instance segmentation. This real-time capability better fulfills the needs of agents when perceiving their surrounding environment.

2.2 Navigation

Based on the map[2-4], implemented path planning for a specific goal location. Target Driven Navigation is to navigate to a given target in an unknown environment[9]. ObjectNav[5] required the agent to move to the target object in an unseen indoor environment. In the ObjectNav challenge, Chaplot et al.[10] and Campari et al.[11] employed Reinforcement Learning (RL) techniques and achieved notable performance, while Zhu et al.[12] improved navigation by predicting the distance to the target using semantically related objects as cues. When faced with map-assisted object search navigation, the problem can be simplified.

2.3 6-Dof grasp

A 6-Dof (Degree of Free) grasp refers to a grasping pose in three-dimensional space, characterized by translation (3-Dof) and rotation (3-Dof). This allows for exceptional precision and flexibility in manipulating objects. With the advancement of deep learning, 6-Dof grasping has seen rapid progress[13]. Mousavian et al.[14] train a variational autoenconde to sample a set of grasps. Fang et al.[15] released the Graspnet-1Billion dataset, a 6-Dof grasping dataset that provides dense annotations for each model through force closure and maps these models to real RGB-D data. Wang et al.[16] proposed a definition of "graspness" to determine grasping points within cluttered environments. However, these works primarily focus on environments with fixed backgrounds such as workbenches, whereas real-life scenarios often involve more complex backgrounds.

3. DESIGN OF THE PCRGN FRAMEWORK

Motivated by the MRGNC, which presents a challenge for robot navigation and grasping capabilities in complex household scenarios, this section proposes a holistic robot grasping and navigation framework based on RGB-D images (namely the PCRGN) that allows the agents to (i) percept the surrounding environment, (ii) search and navigate to the target object, and (iii) grasp the object under complex scenarios. In essence, the constructed framework consists of three major modules: perception, navigation, and grasp, each fulfilling one task listed above. Below we describe the target scenario (Section 3.1), the overall structure of the complete PCRGN Framework (Section 3.2), and the detailed design of each individual module (Sections 3.3-3.5).

3.1 Preliminaries

The MRGNC provides a robot navigation and grasping simulator for a wide range of household scenarios[1], as shown in Figure 1. The scenario includes four distinct rooms (i.e., bedroom, living room, parlor, kitchen), each furnished appropriately, e.g., a bed in the bedroom, and a couch in the living room. The simulator can generate various scenarios with different furniture layouts and object distributions in every room. Additionally, a cost map indicating whether there exists an object in a given position is provided for navigational purposes. The simulator adopts a dual-wheeled robot, equipped with a robotic arm and two RGB-D sensors-one positioned at the end of the arm and the other on the head of the robot, including basic sensing, movement, and arm control functionalities, providing the foundation for the interaction between the robot and the surroundings.

Target problem. Based on the scenario, the task is to navigate the robot to identify and move the target object to a predetermined spatial location. For instance, the robot can be instructed to navigate into the bedroom, search and grasp a can, and then place it in a specified basket in the living room. To accomplish the task, navigation through different rooms to identify the target object and object grasping is required.

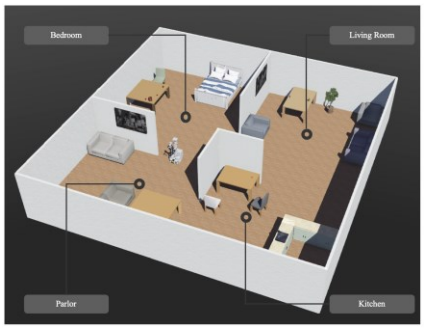

Figure 1. An example scene of MRGNC from[1].

3.2 Overall structure of PCRGN

In household scenarios, navigation requires the location information of objects while grasping needs information about the shapes of the target object. The success of both tasks is based on the accurate recognition of the target object. To achieve this, we propose a Point Cloud-Centric Robotic Grasping and Navigation (PCRGN) framework, composed of the following major modules.

- A **perception** module that provides accurate spatial information of the target object, allowing precise navigation and grasping;
- A **search** module that navigates the agent to the target object;
- A **grasp** module that generates a 6-Dof grasping pose for the target object.

Agent states. In addition, to decide which module should be invoked at different execution stages of the agent, the following agent states are defined: navigation state, grasping state, and placement state (moving the target object to a pre-defined position). Among these states, the navigation and placement would involve the search module, whereas the grasp module is invoked during the grasping state.

Working process of PCRGN. Figure 2 presents a complete working process of the constructed PCRGN framework. In general, the PCRGN takes the RGB-D images and the state of the agent (i.e., navigation, grasping, or placement) as the input, and determines the next action of the robot, e.g., moving to a position or grasping the object. Upon each input information, the perception module first extracts the category and spatial information of objects from the RGB-D images. Depending on whether the target object is in the input image, the agent enters into the navigation state to (i) keep searching for the object or (ii) move to the target object. Then, the search module is applied to calculate the next position of the agent through the spatial information obtained from the perception module. When the object is found, the agent enters into the grasping state. The grasp module predicts a grasping pose based on the spatial information from the perception module. After the grasping, the state of the agent is updated to the placement, in which it invokes the search module to find the pre-defined position to place the object. Below we detail the design of each module in the PCRGN framework.

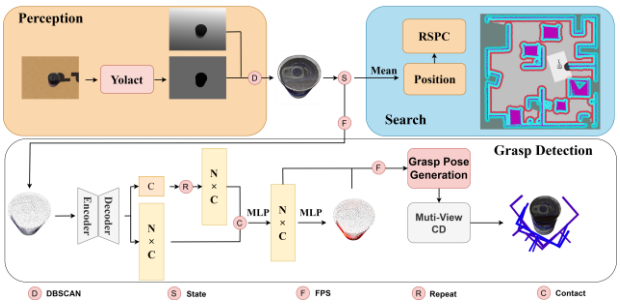

Figure 2. Method structure.

3.3 Perception module

The perception module is a key part of this framework, providing essential environmental information for the search and grasp modules. To achieve this, the point clouds are extracted from the objects in the environment to represent three-dimensional information, providing fine-grained and highly-salable environmental perception that can facilitate the searching and grasping process, respectively. With the point clouds obtained, the navigation module can get accurate location information of objects and the grasp module can get information about the shapes of the objects in the environment. However, when observing long-distant objects, it is difficult for the point clouds to accurately segment objects due to sparse pixels. As objects often exhibit distinct features such as color and texture in comparison to the background, we have opted to use image-based segmentation. By utilizing the alignment between RGB and Depth images, we can pull point cloud data from the segmented areas on RGB images. With point clouds and image-based segmentation, the robot is able to perceive the surrounding environment and extract the information needed by navigation and grasp. The detailed design and implementation of the perception module are described in Section 4.1.

3.4 Search module

The purpose of the navigation is to control the agent to find and approach the target object. In this work, the NRGNC[1] provides the path planning algorithm. This allows us to simplify the navigation task to (i) the computation of the target object position and (ii) the deduction of an appropriate robot pose towards the object.

In household scenarios, it is common to find objects on counters, tables, and similar furniture. Therefore, if the target object is not identified immediately without moving the robot, it would be beneficial to consider searching for objects that are commonly found nearby.

Based on the above, we propose a search module that inputs point clouds to get the location of objects. In addition, to facilitate the search task, we propose an algorithm named Robot Search based on the Point Cloud (RSPC) that finds the target object using associated objects (e.g., tables and counters) as clues, improving the effectiveness and success ratio of the searching process. The detailed implementation of the search module is detailed in Section 4.2.

3.5 Grasp module

In household scenarios, the relative positioning of agent with objects can be complex, and the agent may observe objects from any angle. In addition, objects often have different poses and backgrounds. This imposes a significant challenge on object grasping using a mobile robotic arm. To address this, data-driven deep learning 6-DoF grasping solution, trained on diverse datasets, is constructed to effectively handle the complex relationships between the robotic arm and the target object. In addition, a dataset collection scheme is proposed based on the MRGNC simulation environment to facilitate the construction of the training dataset, effectively improving the diversity of the training data and, hence the quality of the model. Also, MRGNC aims not just to grasp objects but to do so optimally, evaluating 6-Dof grasp quality through force closure and contact area. Existing deep-learning-based grasping methods usually focus only on the success or force closure of the grasp[14,15,17]. We define the grasp label empirically based on the MRGNC simulator.

For accurate and robust grasps[16], proposes a definition named graspness as the quality metric to determine where to grasp the target object in cluttered scenes. In this work, the method in Reference[16] is applied as the foundation for object grasping. In addition, a set of problem-specific optimizations are conducted on the method in Reference[16], which explicitly considers the characteristics of the target objects (i.e., single household items), effectively improving the effectiveness of the proposed framework. The details of the grasp module are presented in Section 4.3.

4. FRAMEWORK IMPLEMENTATION

This chapter details the implementations of the three modules in the PCRGN Framework, along with the corresponding data collection strategies.

4.1 Perception module

4.1.1 Model design. Given the RGB-D images, we use a semantic model to get the semantic segmentation about RGB and get the point cloud of the desired objects by aligning it with the depth map. Given the frequent invocation of the visual module during operations, the segmentation model must maintain minimal time overhead. We have chosen the YOLACT mode[18] as the segmentation tool for its impressive real-time segmentation capabilities.

Semantic segmentation of RGB images can still introduce noise at the edges of objects. This noise can lead to significant errors in location information, especially when the target object occupies a small portion of the image. To minimize this error, we perform clustering on the point clouds obtained from the segmented images to exclude outlier points, as shown in Figure 3.

(a) RGB (b) Yolact (c) Ours

Figure 3. Visualizetion of perception module.

4.1.2 Dataset prepartion. In the MRGNC's easy mode, an interface is provided to obtain semantic segmentation maps from the environment. We direct the robot to random locations within a specific room in the simulated environment to gather the necessary dataset for training the segmentation model. By positioning the robot at random locations and rotations, we capture target object data (RGB images and segmentation maps), automating data collection and enhancing the dataset's generalizability.

To this end, we constructed the perception model of the PCRGN framework that utilizes RGB-D data to extract target object point clouds and trained it with data collected in the MRGNC simulation environment.

4.2 Object searching

4.2.1 Search algorithm. The critical steps in object search involve (i) getting the position of the target object and (ii) steering the robot to an accessible location near the object. In household settings, objects that robots can directly grasp are typically placed on tables, counters, and similar surfaces. Drawing inspiration from this, when the robot can't directly observe the target object, it can approach the vicinity of related objects to find the target object. To implement this, we've developed a novel algorithm, named RSPC, that computes the possible next steps for the agent based on the locations of relevant objects, as illustrated in Algorithm 1. A comprehensive description of the proposed positional function and the RSPC algorithm is provided below.

Initially, we define the related object as r={c, pc, v, δ, P}, where C denotes the object category, pc refers to its point cloud, v indicates the object visitation status, δ denotes the minimum distance between of agent and object that allows the agent to have a better visual field and prevent potential collisions, and P denotes object's priority. The values of c, δ, and P are predefined for each object, with v set to false initially and P is none.

For relevant object sets R={r||i=1, ..., N}, whenever ri in R is found by perception module, ri .pc is updated (Line 3 in Algorithm 1). Before the agent requires transitioning to a new position, the object in R with the highest priority P, Pc is not none, and a visitation status V of false will be selected (Line 4 in Algorithm 1). The agent approaches a location near it that has not been visited if a new relevant object is not yet discovered. Upon detecting a relevant object, the agent will navigate to a location close to this object (as outlined in lines 5-17 of Algorithm 1). However, even after determining the target position, obstacles and potential collisions might prevent direct access. To address this, we use the identified position and a δ margin to compute a feasible pose the agent can reach.

Calculate_pose Function. This function determines a pose closest to the target position where the surrounding area within a δ range in the cost map is free of obstacles, indicated by a value of 0. Then agent toward the position input.

4.3 Object grasping

4.3.1 Dataset preparation. Grasp datasets can annotate grasp labels for each object model and then transfer the labels to each specific scenario through coordinate transformations. The following presents our design philosophy, which includes the object model and grasp labels, scene data, and feature labels.

Object Model and Grasp Label. The GraspNet-1Billion[15] encompasses model information for the objects we require, along with abundant grasp labels. However, its grasp labels only consider the force closure metric and do not align with the assessment criteria for grasp quality in this competition. Hence, we have redefined the label through equation (1).

$$(C = 1)\left[\alpha \min\left(\frac{S_n}{S_t}, 1\right) + \beta(2 - S_{fc})\right] \tag{1}$$

where C is the collision detection metric, which returns 0 if grasp poses results in a collision with the object; S_t is the maximum number of valid points; S_{fc} is a force closure metric, which assesses whether there exists a set of forces that can maintain a stable and closed state of the fingers while gripping an object, providing sufficient torque to prevent the object from sliding or slipping out of the grasp; S_n is the valid point in the gripper closure area. If the angle between the gripper approach vector and the point normal is less than 30 degrees, it will be counted as a valid point.

Feature Label. The concept of graspness proposed in GSNet[16] only considers whether a successful grasp can be achieved, neglecting the impact of grasp quality. We extend its application to the environment of grasp quality through quality-graspness for each object model by equation (2).

$$\mathrm{S} = \left\{ \mathrm{s_i} \middle| \mathrm{s_i} \frac{\sum_{v=1}^{V} \sum_{a=1}^{A} \sum_{d=1}^{D} \mathrm{q}(\mathrm{g_{vad,i}})}{V*A*D} \right\} \tag{2}$$

Algorithm 1 Robot Search based on the Point Cloud (RSPC)

Output: Go to the target object

```
flag=Flase;
while robot not reach final pose do
    rotate the robot to obtain environmental information;
    r=Get a relevant object()
    if r is None then
        the current position in the visited map set 1;
        get the nearest position from that visited map with
        a value of 0;
        δ=0;
    else
        Position = mean of r.Pc;
        δ=r.δ;
        r.v=Ture;
    end if
    if r.C is target object then
        flag=True;
    end if
    target_pose=Calculate pose (Position, δ);
    go to target pose;
    the current position in the visited map set 1;
    if flag then
        break;
    end if
end while
```

in which i is the ith point of a model, V is the number of views, A is the number of angles, D is the number of widths, $g_{vad,i}$ is the grasp base on ith point generated by v, a, d. Function q() returns the score of the grasp, which is calculated by equation (1).

Finally, to obtain a coherent representation of the object-level scores, we perform a normalization for each object, as shown in equation (3).

$$S = \frac{S - Min(S)}{Max(S) - \mathrm{Min(S)}} \tag{3}$$

Scene Information. To gather scene information, we leverage the aforementioned object-searching algorithm to locate the target object in the simulated environment. The object is then observed from multiple perspectives, from which we collect RGB images, segmentation maps, and depth maps, as well as the pose information of both the object and the camera. For each data, the grasp labels mentioned above are mapped onto this data through coordinate transformation. Then we can generate collision labels through collision detection.

4.3.2 Grasping model. To better adapt the method GSNet to our framework, we made the following optimizations concerning its input data and network model.

Data Preprocessing. The method GSNet is designed for grasping in cluttered scenarios, and it reduces the input quantity through random sampling of all acquired point cloud data. However, when the background is complex, this method may struggle to distinguish between the target object and the background. In the case of known object grasping, we can adopt the FPS sampling method after extracting the point cloud information of the target object through the perception module. This not only significantly reduces the noise but also better retains the object information. Eventually, we sample N points to form an N∗3 input for the network model.

Network Model. To capture local and global features, we employ the ResUNet14 based on MinkowskiEngine[18] to generate C dimensional features for each point and C dimensional global features. We then merge these features through the repeat and concat. We use the MLP to regress the fused features to high-dimensional features N∗C for subsequent grasp label regression. This high-dimensional feature regresses the quality-graspness for each point through the MLP, and after threshold filtering of quality-graspness (N∗1), FPS is used to obtain feature points for grasp post generation. The grasp post is obtained using the grasp post generation method in GSNet.

Muti-View Collision Detection. By fusing point clouds generated by the perception module from multiple perspectives, we complete some of the object model information and perform collision detection with the grasping posture generated by the aforementioned model. Simultaneously, we filter out some of the bottom-up grasp posts by limiting the angle of the grasping posture.

5. EXPERIMENTS

5.1 Experimental setup

MRGNC provides 80 scenes for training and validation. We adopt the method in Section 4 to get the training data. For the perception module, we collect 400 images for each object following the format of COCO[19]. We train on GTX-1050Ti, batch size 64, the learning rate is 0.00001, and epoch is 200. For the grasp module, we collect 200 pieces of data for each object following the format of GraspNet[15]. We train it on Tesla-V100, batch size 4, the learning rate is 0.001, and the epoch is 18.

Below is the method of testing the capacity of our framework provided by[1]: MRGNC has 20 non-public scenes for testing. There are four tracks in this challenge, including normal track, easy track, grasp detection track, and object search track. For the normal channel, it will automatically evaluate the result of the normal, grasp detection and object search track, and is unable to use the ground truth segmentation result. For the Easy channel, the ground truth segmentation map can be obtained by socket API, and the result of the easy track and grasp detection track will be automatically evaluated. In each scenario, we are tasked with the following sequence: “Go to Room A, grasp Object B, and place it in Room C.” The testing contains three partitions:

- **Object search.** From the moment the task is issued, the goal is to navigate to the vicinity of Object B.
- **Grasp detection.** After object search, the algorithm needs to generate grasp poses, and the server will score this grasp pose.
- **Placed.** Form successes grasp the object to arrive at the table in room C.

The time limit of the task is 10 minutes, object search score and placed score are given by equation (4).

$$\left(1-\frac{T}{600}\right)\times\alpha \tag{4}$$

For the object search score, α is set to 40 for easy and normal tracks and set to 100 for the object search track. For the placed score, α is set to 10.

The grasp detection score is given by equation (5).

$$\frac{1}{N}\sum_{i=1}^{N}(C=1)\left[14\min\left(\frac{S_n}{500},1\right)+20(2-S_{fc})\right] \tag{5}$$

N is the number of detected grasp poses, which is less than 10. C is the collision detection metric, which returns 1 if the grasp pose occurs in collision with the object. S_n is the valid point in the gripper closure area. If the angle between the gripper approach vector and the point normal is less than 30 degrees, it will be counted as a valid point. S_{fc} is a force closure metric. For grasp detection track, the grasp detection score will be doubled.

The final score is the summation of object search, grasp detection, and placed scores.

$$S = Ssearch + Sgrasp + Splaced \tag{6}$$

5.2 Test result

A series of experimental tests are carried out to evaluate the performance of our framework in the domains of normal, easy, grasp, and search. The results of these experiments quantified through scores are shown in Table 1.

Table 1. The evaluate results of MRGNC.

	Easy	**Normal**	**Grasp**	**Search**
Baseline	30.32	-	3.57	-
Ours	79.5	82.4	76.4	90.8

Our framework gets the first price of MRGNC. We have achieved a 160% improvement over the baseline in the easy tracks and our score was 70 points higher than the baseline in the grasp track. These results indicate that in this environment, our method can more quickly locate items and grasp them with higher quality.

We visualize some grasp generated by our approach and graspness[16] when have a complex background, as shown in Figure 4. The redder the grasp pose, the higher the score; conversely, the bluer the pose, the lower the score. Our method demonstrates better performance for grasps applied to objects. Additionally, when faced with complex backgrounds, our method effectively filters out background noise, ensuring that the grasp is more accurately focused on the objects.

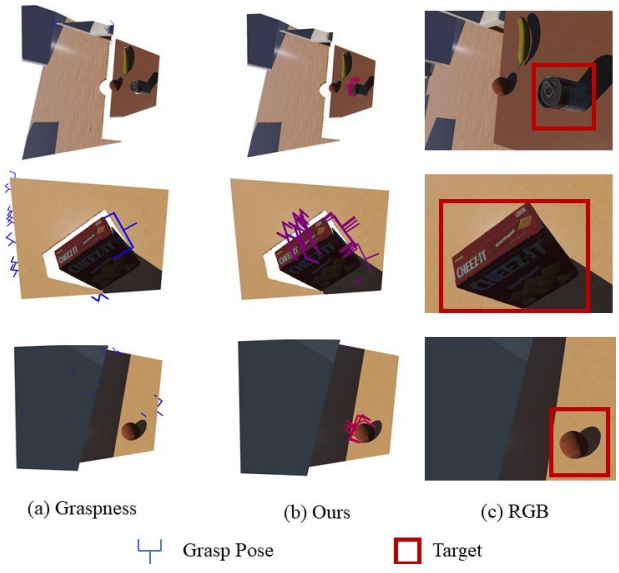

Figure 4. Visualizetion of grasp on complex background.

6. CONCLUSION

We proposed a framework composed of three components (search, perception, grasp) specifically designed for navigation and grasping tasks in household scenarios. In the MRGNC2023, our framework successfully completed all test tasks and achieved the highest scores in the challenge. However, it still lacks application in real-world scenarios, which will be our focus for future work.

REFERENCES

[1] "Mobile robot grasping and navigation challenge 2023," TCRI, Bytedance AI Lab, <https://homepage.robotmani-challenge.com> (1 June 2023).

[2] Sethian, J. A., "A fast marching level set method for monotonically advancing fronts," Proceedings of the National Academy of Sciences 93(4), 1591-1595 (1996).

[3] Hart, P. E., Nilsson, N. J. and Raphael, B., "A formal basis for the heuristic determination of minimum cost paths," IEEE Transactions on Systems Science and Cybernetics 4(2), 100-107 (1968).

[4] Karaman, S. and Frazzoli, E., "Sampling-based algorithms for optimal motion planning," The International Journal of Robotics Research 30(7), 846-894 (2011).

[5] Batra, D., Gokaslan, A., Kembhavi, A., Maksymets, O., Mottaghi, R., Savva, M., Toshev, A. and Wijmans, E., "Objectnav revisited: On evaluation of embodied agents navigating to objects," arXiv, (2020).

[6] Minaee, S., Boykov, Y., Porikli, F., Plaza, A., Kehtarnavaz, N. and Terzopoulos, D., "Image segmentation using deep learning: A survey," IEEE Transactions on Pattern Analysis and Machine Intelligence 44(7), 3523-3542 (2021).

[7] He, K., Gkioxari, G., Doll'ar, P. and Girshick, R., "Mask r-CNN," Proceedings of the IEEE International Conference on Computer Vision, 2961-2969 (2017).

[8] Bolya, D., Zhou, C., Xiao, F. and Lee, Y. J., "Yolact: Real-time instance segmentation," Proceedings of the IEEE/CVF International Conference on Computer Vision, 9157-9166 (2019).

[9] Zhu, Y., Mottaghi, R., Kolve, E., Lim, J. J., Gupta, A., Fei-Fei, L. and Farhadi, A., "Target-driven visual navigation in indoor scenes using deep reinforcement learning," IEEE International Conference on Robotics and Automation, (2017).

[10] Chaplot, D. S., Gandhi, D., Gupta, A. and Salakhutdinov, R., "Object goal navigation using goal-oriented semantic exploration," Neural Information Processing Systems (NeurIPS), (2020).

[11] Campari, T., Eccher, P., Serafini, L. and Ballan, L., "Exploiting scene-specific features for object goal navigation," European Conference on Computer Vision, 406-421 (2020).

[12] Zhu, M., Zhao, B. and Kong, T., "Navigating to objects in unseen environments by distance prediction," 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 10571-10578 (2022).

[13] Newbury, R., Gu, M., Chumbley, L., Mousavian, A., Eppner, C., Leitner, J., Bohg, J., Morales, A., Asfour, T., Kragic, D., et al., "Deep learning approaches to grasp synthesis: A review," IEEE Transactions on Robotics 39, 3994-4015 (2023).

[14] Mousavian, A., Eppner, C., and Fox, D., "6-dof graspnet: Variational grasp generation for object manipulation," Proceedings of the IEEE/CVF International Conference on Computer Vision, 2901-2910 (2019).

[15] Fang, H. S., Wang, C., Gou, M. and Lu, C., "Graspnet-1billion: A large-scale benchmark for general object grasping," Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 11444-11453 (2020).

[16] Wang, C., Fang, H. S., Gou, M., Fang, H., Gao, J. and Lu, C., "Graspness discovery in clutters for fast and accurate grasp detection," 2021 IEEE/CVF International Conference on Computer Vision (ICCV), 15944-15953 (2021).

[17] Sundermeyer, M., Mousavian, A., Triebel, R. and Fox, D., "Contact-graspnet: Efficient 6-dof grasp generation in cluttered scenes," IEEE International Conference on Robotics and Automation, (2021).

[18] Choy, C., Gwak, J. and Savarese, S., "4D spatio-temporal convnets: Minkowski convolutional neural networks," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 3075-3084 (2019).

[19] Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Doll´ar, P. and Zitnick, C. L., "Microsoft coco: Common objects in context," Computer Vision-ECCV 2014: 13th European Conference, Zurich, Switzerland, 740-755 (2014).