2017 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)
Sheraton Arabella Park Hotel, Munich, Germany, July 3-7, 2017

Object Placement Estimation with Occlusions and Planning of Robotic Handling Strategies

Wataru Miyazaki and Jun Miura
Department of Computer Science and Engineering
Toyohashi University of Technology

***Abstract*— This paper describes a system that can find and lift a specific object in a bin containing piled objects. Such a task is ubiquitous in our daily life, for example, in finding a small toy in a toy box or in finding a stationary in a drawer. To efficiently achieve this task, it is necessary to recognize the object placements with consideration of occlusions and to plan a proper hand motions for lifting or searching for the target object. We developed methods for such two necessary functions, with introducing a sweep motion for removing many non-target objects at once. We implemented the methods on a dual-arm humanoid robot with an RGB-D camera and a suction mechanism. The experimental results show the effectiveness of the proposed approach.**

***Index Terms*— Object search under occlusion, Object handling, Object placement recognition, Bin-picking**

I. Introduction

Personal service robot is one of the promising application areas of robotic technologies. In the "aging society," the expectations of robots that can support people in their everyday situations are increasing. Possible tasks of such robots are: fetching a user-specified object, putting tableware away, and cleaning a room. Object search and manipulation is one of the commonly-used functions for such robotic tasks.

In this paper, we deal with the task of finding a specific target object in a bin with many objects, where the target object is sometimes occluded by others. We often face such a situation in our daily life (see Fig. 1 for example), and our usual strategy will be something like localizing the object from its partially-occluded view, picking up a most likely object, and/or stirring the object pile, for getting the object. This paper aims to develop a system that can handle such situations in the task of finding a specific object.

Hand-eye systems have a long history and their functions can usually be divided into two parts: (1) object recognition and localization and (2) hand motion planning.

Fig. 1. Typical cases of finding an object with occlusion.

Typical strategies for object recognition and localization are shape-based [1] and feature-based [2], [3]. They use object shape and appearance (or texture) models and try to match them with those in the scene. For texture-scarce objects, edge-based features have been proposed [4], [5]. Feature-based approaches are basically robust to partial occlusions as long as prominent local features are visible. It is also possible to localize an object from visible features by, for example, solving PnP problem [6].

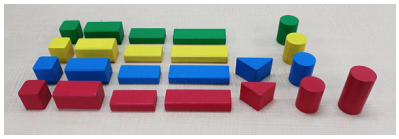
In the case where such strong features are not available and/or under heavy occlusions, recognition results should include ambiguities. For cases where occlusions exist, Dogar et al. [7] developed a method of evaluating occluded regions behind observable objects and predicting possible existence of a target object in the regions. Since all objects are separated and stand on the table, a simple strategy of removing the largest object first is sufficient for finding an occluded target object.

Hand motion planning has also been investigated for a long time. Typically, it is solved by determining hand pose for grasping [8] and planning a collision-free hand motion; solutions are usually straightforward once the object pose is reliably estimated. Recent works on pick-and-place planning deal with more realistic problem setting, for example, complex object and environment surface [9], [10] and human-like hands and re-grasping [11].

In a complex scene as shown in Fig. 1, integration of recognition and handling is sometimes effective. Gupta and Sukhatme [12] use pushing actions to spread a pile or a cluster of objects so that each object can be recognized and manipulated easier. Recognition is done only when objects are sufficiently apart from each other. Katz et al. [13] adopt pushing operations for verifying object segmentation hypotheses in clearing a pile of unknown objects. Spending time for recognizing a scene as accurately as possible could be worse than just handling objects one after another with a rough interpretation of the scene. Balancing recognition and action operations is an interesting issue.

This paper deals with a probabilistic interpretation of object placement in a scene with a heavy occlusion, followed by an adaptive object handling strategy; depending on the interpretation, that is, how reliably a target object is found and localized, an appropriate handling action is selected. The contribution of the paper is to develop an integrated approach to object search in a clutter, with introducing a sweep motion

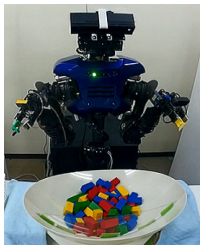
978-1-5090-5999-7/17/$31.00 ©2017 IEEE



(a) The set of blocks used.

(b) The container from which a target object is lifted.

Fig. 2. Bin-picking task.



(a) HIRO.

(b) Suction nozzle.

(c) Plate for sweep.

Fig. 3. The robot with a suction mechanism.

to remove many non-target objects at once, and to verify the approach in a real implementation.

The rest of the paper is organized as follows. Section II describes an overview of the system and the robot and the tasks we deal with. Section III describes the details of an object placement estimation method. Section IV describes a method of planning handling strategies. Section V concludes the paper and discusses future work.

II. Overview of the System

A. Task and robot system

We deal with the task of finding and lifting a target object from a set of known objects (blocks) in a clutter as shown in Fig. 2. The shape of the container is known to be a part of a sphere, and its size and position are measured in advance.

Fig. 3 shows our robot (HIRO of Kawada Robotics) with a Kinect V2 on the head for image and depth acquisition. A suction mechanism and a plate are attached to either of the hands for lifting an object and sweeping objects, respectively.

B. Process flow

The robot takes the following steps for achieving a task:

1) Estimate object placements by extracting surfaces, enumerating possible correspondence and object model candidates, and calculate the probability of each scene object matching with an object model.
2) Lift an object using the suction mechanism by planning a collision-free hand motion if target object candidates are found.
3) Verify the lifted object by checking its two different surfaces.
4) Estimate the size of occluded regions for planning a sweep motion if target object is not found.

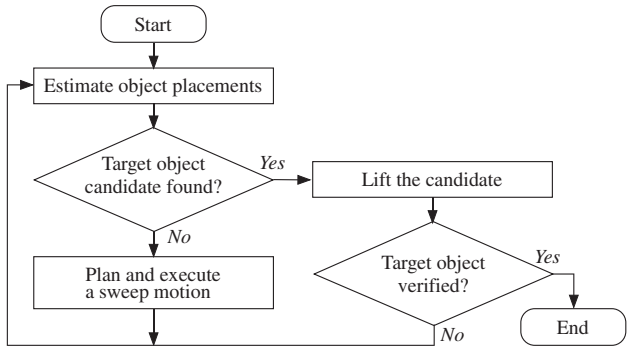


Fig. 4. Flow of target object finding process.

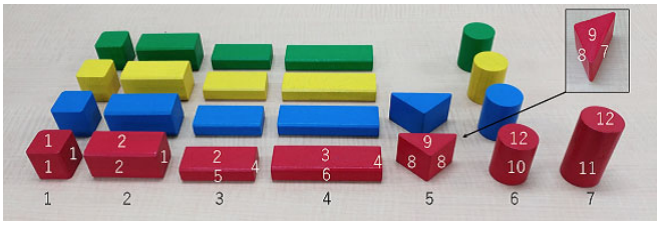


Fig. 5. Object models and surface models.

5) Sweep the selected region for hopefully revealing the target object.

The flow of the object finding process is summarized as shown in Fig. 4.

III. Object Placements Estimation

Object recognition and object localization are necessary for lifting a target object. Due to frequent occlusions, the target object may not be visible or may be only partially visible. Since ignoring partially-visible objects in a very cluttered scene as shown in Fig. 2(b) is inefficient, we infer object identities also for such objects.

An object is composed of surfaces. We thus first detect surfaces in a scene, and then estimate the probability of each surface being a certain model surface. We then calculate the probability of an object in the scene being an object model in the database using object-surface relationships. The details are explained below.

A. Object and surface models

The database describes the relationships between objects models and surface models. There are twelve surface models for seven object models, as shown in Fig. 5. Cylindrical surfaces (surface #10 and #11) are sometimes detected as a pair of planes due to a limited accuracy of KinectV2. We add two virtual surface models (#13 and #14), which are observed in such a case, and also two virtual object models composed of those planes.

B. Surface detection and identification

The first step is to detect surfaces using an HSV clustering and a RANSAC-based multiple surface fitting [14]. The fitting is performed for both flat and curved surfaces. Fig. 6 shows an example surface detection result.

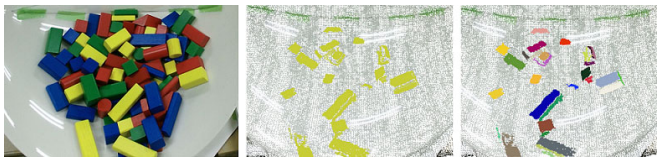
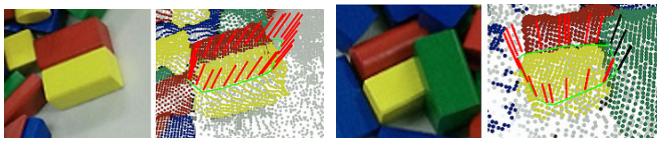


Fig. 6. An example result of color-based pixel clustering and surface detection. From left to right: KinectV2 input, detected only yellow objects, and surface detection result.



(a) Un-occluded case. (b) Occluded case.

Fig. 7. Occlusion status check. Red lines indicate unobstructed views and black ones obstructed views.

For identifying a surface, we treat un-occluded and occluded cases differently. When a surface is judged to be un-occluded, the generic surface shape models (rectangle, triangle, circle for planar, and cylindrical for curved in our case) are fitted to the surface. Whether a surface is occluded or not is judged by checking the existence of other objects between the camera position and the region surrounding the surface boundary. Fig. 7 shows an occluded and an un-occluded case judged by this check.

1) Surface identification for un-occluded cases: In the surface model fitting for a detected planar surface, the minimum bounding shape of each generic surface shape model is calculated. Fig. 8 shows the case where three generic planar surface models are fitted to a detected rectangular surface. If the ratio of the area of the detected surface to that of a fitted model is above a threshold (currently, 0.7), this fitting is accepted and two model size parameters (e.g., the lengths of two rectangle edges) are calculated. Possible *specific* models in Fig. 5 are then enumerated for which the differences in size parameters are less than another threshold (currently, $10.0\,mm$).

The probability $P(S_m, S_d)$ that a detected surface S_d corresponds to a model surface S_m is given by:

$$P(S_m, S_d) = \alpha M(E_m^1, E_d^1) \cdot M(E_m^2, E_d^2), \tag{1}$$

$$M(E_m, E_d) = \begin{cases} \min\{E_m/E_d, E_d/E_m\} \\ \qquad (|E_m - E_d| < 10\,mm) \\ 0 \quad \text{(otherwise)} \end{cases} \tag{2}$$

where E_* is an edge length and α is a constant for normalization over possible surface models.

2) Surface identification for occluded cases: Identifying an occluded surface is sometimes hard only from the visible part. We thus enumerate surface models which are *not inconsistent* with the detected ones in terms of the maximum and the minimum size and give a uniform probability to each model, that is,

$$P(S_m, S_d) = \frac{1}{N}, \tag{3}$$

where N is the number of enumerated models.

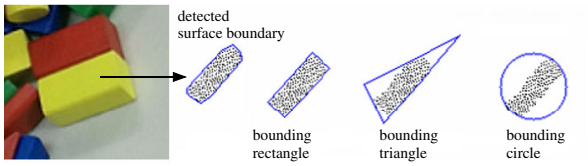
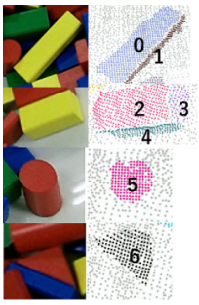


Fig. 8. Generic surface model fitting.



Detected surface	Candidate model surfaces (probability)
0	3(1.000)
1	6(1.000)
2	2(0.632),7(0.368)
3	1(0.322),7,(0.273),8(0.405)
4	2(0.744),7(0.256)
5	1(0.463),8(0.146),12(0.391)
6	1(0.166),2(0.166),3(0.166),7(0.166),8(0.166),9(0.166)

Fig. 9. Surface identification results.

3) Example surface identification: Fig. 9 shows examples of surface identification. Surfaces 0 to 5 are un-occluded and surface 6 is occluded. Surfaces 0 and 1 are identified as a single model because their sizes are large enough for discrimination. Surfaces 2 to 5 are matched with multiple models because their sizes are not very unique. Surface 6 has many possible models due to occlusion.

C. Object identification from surface identities

Object identity can be determined from those of detected surfaces. We first cluster neighboring surfaces into objects if they are close enough ($< 10\,mm$) to each other and their relative angle is small enough ($< 90\,deg$). Fig. 10 shows a result of clustering for yellow blocks.

For a detected object with one surface S_d, the probability $P(O_m, S_d)$ that it belongs to model object O_m is given by:

$$P(O_m, S_d) = \beta \sum_i^{N_s} P(S_m^i, S_d) \cdot P(O_m, S_m^i), \tag{4}$$

where N_s is the number of model surfaces and β a constant for normalization over possible object models. The first term is the probability that detected surface S_d corresponds to model surface S_m^i, defined above. The second term is the probability that model surface S_m^i belongs to model object O_m. This is given by dividing the number of that surface in the model object by the number of that surface over all model objects. As a result, when surface 2 is detected, for

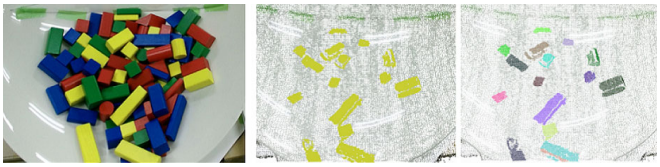


Fig. 10. An example result of clustering surfaces. From left to right: KinectV2 input, detected yellow surfaces, object detection result; each region of a color corresponds to an object.

Detected object	Candidate model objects (probability)
0	3(1.000)
1	2(0.649),5(0.351)
2	1(0.346),2(0.116),3(0.146),6(0.196),7(0.196)
3	1(0.125),2(0.151),3(0.055),4(0.166),5(0.498)

Fig. 11. Object identification results.

example, this probability is larger for object 2 than for object 3, since object 2 has two of surface 2 (see Fig. 5).

For a detected object with two surfaces, S_d^1 and S_d^2, the probability $P(O_m, S_d^1, S_d^2)$ that it belongs to model object O_m is given by:

$$P(O_m, S_d^1, S_d^2) = \gamma \sum_i^n \sum_j^n P(S_m^i, S_d^1) \cdot P(S_m^j, S_d^2) \cdot P(O_m, S_m^i, S_m^j), \quad (5)$$

where γ is a constant for normalization over all combinations of model surfaces. The first and the second term are the probability about the surface identity defined above. The third term is the probability about surface-object relationships, defined accordingly as in the case of single surface-detected objects.

Fig. 11 shows examples of object identification. Objects 0 and 1 have one or two candidate models. Objects 2 and 3 have much more candidates because only one surface is detected.

D. Experimental evaluation

We experimentally compared the proposed object placement estimation method with a method which does not consider occlusions. We put all objects in the container and stirred them sufficiently, and then executed the two estimation methods. For each model object, we collected thirty un-occluded cases and thirty occluded cases, by repeatedly performing the above steps (i.e., put, stir, and estimate steps).

The comparison results are summarized in Table I. While both methods exhibit very similar performances in un-occluded cases, ours largely outperforms the other in the occluded cases. Although the recognition sometimes fails even if we take top-three ranks, we can cope with such a situation by planning a robust handling strategy, as explained in the next section.

IV. Planning Object Handling Strategies

The robot tries to lift a target object when it is found by the object placement estimation. If the target object is not found, the robot takes an action to remove some objects so that objects underneath will be visible, hoping the target object is included there. In this paper, we introduce sweep motions which can remove multiple objects at once.

TABLE I

Comparison of object identification methods with/without considering occlusion.

(a) Number of correctly identified objects at top rank in un-occluded cases.

Methods \ Object ID	1	2	3	4	5	6	7
Proposed	21	7	10	23	22	12	17
No occlusion-aware	21	7	14	23	22	10	16

(b) Number of correctly identified objects at top rank in occluded cases.

Methods \ Object ID	1	2	3	4	5	6	7
Proposed	9	4	10	22	27	4	13
No occlusion-aware	6	0	8	4	15	7	5

(c) Number of correctly identified objects within top-three ranks in un-occluded cases.

Methods \ Object ID	1	2	3	4	5	6	7
Proposed	27	24	23	23	23	12	17
No occlusion-aware	28	22	23	23	22	10	16

(d) Number of correctly identified objects within top-three ranks in occluded cases.

Methods \ Object ID	1	2	3	4	5	6	7
Proposed	9	18	17	26	27	10	13
No occlusion-aware	12	4	20	14	15	10	5

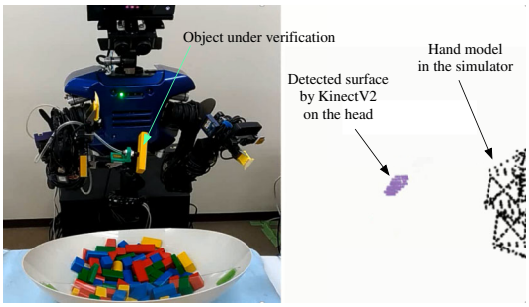


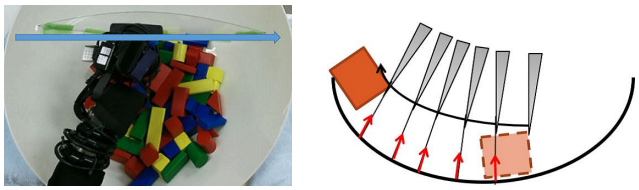
Fig. 12. Verification of the lifted object.

A. Lifting a visible target object

The object placement estimation provides a set of target object candidates with probabilities. If the highest probability is above a threshold, the target is considered to be found and will be lifted up by the robot.

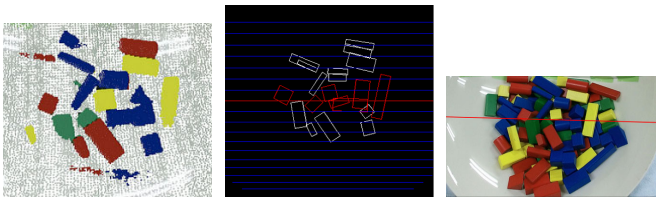
The robot lifts the target object using the suction mechanism. The most important condition for a successful lift is that the tip of the suction nozzle is aligned to the normal of the surface to be lifted. Since there remains an unconstrained degree of freedom around the normal, we prepare a fixed set of angles for that d.o.f. in advance, and test them by checking if a collision-free hand motion is generated.

Once a feasible hand motion is generated, the robot takes the motion to touch the suction nozzle to the surface and starts a suction. After lifting a hand a little, the robot examines if the object is certainly sucked by checking the magnitude of the air pressure. If sucked (i.e., sufficiently low pressure is observed), the robot verifies the object by observing two different surfaces to see if they correspond to those of the model of the target object. Fig. 12 shows a verification scene. If the object is not the target, the robot puts it at a place different from the container and estimates the object placement again.



(a) Sideways motion. (b) Align to the bottom.

Fig. 13. Sweep motion model.



(a) Objects above the sweep motion. (b) Objects to be removed by the motion. (c) The selected motion by Criterion 1.

Fig. 14. Sweep motion planning by criterion 1.

B. Planning a sweep motion

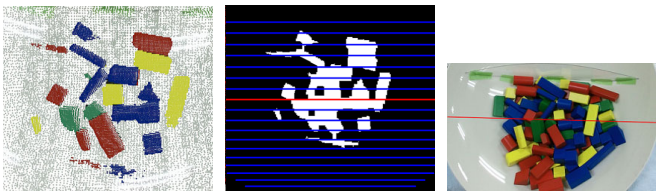
The variety of sweep motions is huge and it is difficult to consider all possible ones. Therefore, we limit the sweeping direction to the sideways and make it align to the bottom of the sphere-shaped container (see Fig. 13). We choose the height of the sweep motion so that it is below by a fixed distance (currently, $15\,mm$) from the top surface of the highest object. We consider and compare the following two criteria for determining a sweep motion.

1) Criterion 1: Maximizing the occluded volume made visible by the sweep motion: This criterion chooses the sweep motion which maximizes the currently-occluded volume to be visible by removing objects on the sweep motion. To this end, the 3D region occluded by each object is calculated as the one between the bottom of the object and the container. The summation of such regions is calculated for each sweep motion candidate, and the one maximizes the sum is chosen. This method requires the placement estimation of all objects in the scene and relatively costly. Fig. 14 shows the selected sweep motion using this criterion.

2) Criterion 2: Maximizing the point cloud volume on the sweep motion: This criterion chooses the sweep motion which maximizes the point cloud volume to be removed by the sweep motion. The volume is calculated just as the sum of point cloud data swept by a motion, and no object recognition is required in this case. Fig. 15 shows the selected sweep motion using this criterion.

C. Experimental evaluation

1) Comparison of criteria for sweep motion planning: We first compared the two criteria for planning a sweep motion. We tested each criterion for cases where the target object is not visible at the beginning in the container with all objects. We evaluate the number of sweep motions needed for making the target object visible and the total time of motion planning and execution. We also evaluated the success rate of finding the target object within ten sweep



(a) Objects above the sweep motion. (b) Point clouds to be removed by the motion. (c) The selected motion by Criterion 2.

Fig. 15. Sweep motion planning by criterion 2.

TABLE II

COMPARISON OF CRITERIA FOR SWEEP MOTION PLANNING.

Method	Ave. # of sweep	Ave. time [s]	Success rate
Criterion 1	3.6	71.28	0.71
Criterion 2	3.7	45.51	0.76

motions. Table II summarizes the results, which show that the criterion to maximize the point cloud volume on the motion (criterion 2) is better, that is, much more efficient and more successful.

2) Comparison of thresholds for identifying the target object: The object placement estimation provides the probability of each detected object being the target object. If the highest one is above a threshold, the target object is considered found and the lifting motion is planned and executed. Changing the threshold will change the robot behavior. That is, a low threshold increases the number of directly lifting the target candidate, but at the same time, increases the verification failure which could increase the total time. A high threshold increases the number of costly sweep motions, but will make the rate of successfully lifting the correct object, thereby decreasing the total time. We would therefore like to seek a good threshold to use.

Table III shows the results for ten rounds for three thresholds. They are compared in terms of the average execution time and the success rate. In a low threshold case, the robot tries to lift a wrong object, which is judged as a most probable object, many times; if the number of this lifting trials exceeds a certain number (currently, 20), this round is considered failure and excluded from the average time calculation. A relatively short average time comes from a few lucky cases where the robot encounters a correct object in an earlier trial. If we continue to the trials until the robot eventually finds a correct object, the average time would be much longer. Based on the result, we chose to use 0.9 as the threshold because its success rate is highest.

3) Comparing Removing Strategy: One possible strategy to search for an occluded target object is to remove one object after another until the target will appear. This does not require a relatively costly sweep motion planning but

TABLE III

COMPARISON OF THRESHOLDS FOR DETERMINING THE TARGET OBJECT.

Threshold	Ave. time [s]	Success rate
0.9	66.08	0.71
0.5	59.68	0.58
0.3	43.12	0.35

TABLE IV

COMPARISON OF REMOVING STRATEGIES.

Method	Ave. # of actions	Ave. time [s]	Success rate
Proposed	4.2	70.13	0.67
Simple	11.8	185.10	0.59

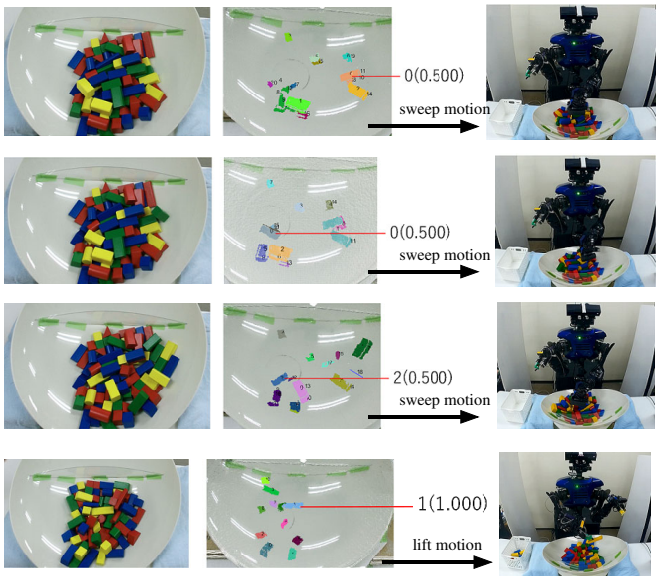


Fig. 16. Steps for successful taking of a target object.

may need to execute removing actions many times. We compare our method with this simple one. Based on the comparisons so far, our method here employs the object placement estimation considering occlusion and the sweep motion planning which maximizes the removed point cloud volume. We also use 0.9 as the threshold for determining the target object.

Table IV shows the results for ten rounds for the two removing strategies. They are compared in terms of the average number of removing actions (i.e., sweeping or lifting actions), the average execution time, and the success rate. Since the execution time for one sweep action and that for one lifting action are similar, executing sweep actions are advantageous because more objects can be removed at a time. This result shows the effectiveness of sweep actions for searching for an occluded target object.

Fig. 16 shows the process of finding a target object (object 4, yellow) by executing three sweep motions and one lift motion. The total execution time was $47.64\,s$.

V. CONCLUSIONS AND DISCUSSION

We have developed a system that can find and pick a target object from a bin with many objects of various kinds. The system employs the two effective methods. One is to estimate object placements statistically considering occlusions and surface-object relationship in the object models. The other is to plan a sweeping motion that can remove as many objects as possible that could be occluding the target object. We implemented these methods on a dual-arm humanoid robot with an RGB-D camera and a suction mechanism, and conducted various experiments. The experimental results show the effectiveness of the proposed methods.

Although the proposed framework is general, the current implementation is for a limited set of objects and for a limited environment (i.e., sphere-shaped container). Adding more objects or introducing a method of automatically generating object and surface models from, for example, CAD models (e.g., [1]) is desirable. In addition, since many objects are in general characterized not only by shape but also by textures, combining shape and texture features in object detection and placement estimation is also future work.

The viewpoint is currently fixed because the RGB-D camera is put on the head of the humanoid. If we put the camera on an arm or if we add a mobility to the humanoid, a more elaborated object placement estimation will be possible by, for example, introducing some viewpoint planning techniques [15], [16]. Integrating such a viewpoint planning into the system could further improve its efficiency.

REFERENCES

[1] K.S. Hong, K. Ikeuchi, and K. Gremban. Minimum Cost Aspect Classification: A Module of a Vision Algorithm Compiler. In *Proceedings of 10th Int. Conf. on Pattern Recognition*, pp. 65–69, 1990.

[2] Z. Ying and D. Castañon. Feature Based Object Recogntion using Statistical Occlusion Models with One-to-one Correspondence. In *Proceedings of 8th IEEE Int. Conf. on Computer Vision*, 2001.

[3] D.G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. of Computer Vision*, Vol. 60, No. 2, pp. 91–110, 2004.

[4] S. Hinterstoisser, C. Cagniart, S. Ilic, P. Sturm, N. Navab, P. Fua, and V. Lepetit. Gradient Response Maps for Real-Time Detection of Textureless Objects. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 34, No. 5, pp. 876–888, 2011.

[5] F. Tombari, A. Franchi, and L. Di Stefano. BOLD Features to Detect Texture-less Objects. In *Proceedings of 2013 IEEE Int. Conf. on Computer Vision*, 2013.

[6] Y. Wu and Z. Hu. PnP Problem Revisited. *J. of Mathematical Imaging and Vision*, Vol. 24, pp. 131–141, 2006.

[7] M.R. Dogar, M.C. Koval, A. Tallavajhula, and S.S. Srinivasa. Object Search by Manipulation. *Autonomous Robots*, Vol. 36, No. 1, pp. 153–167, 2014.

[8] J.L. Jones and T. Lozano-Pérez. Planning Two-Fingered Grasps for Pick-and-Place Operations on Polyhedra. In *Proceedings of 1990 IEEE Int. Conf. on Robotics and Automation*, Vol. 1, pp. 683–688, 1990.

[9] D. Berenson, R. Diankov, K. Nishiwaki, S. Kagami, and J. Kuffner. Grasp Planning in Complex Scenes. In *Proceedings of 2007 IEEE-RAS Int. Conf. on Humanoid Robots*, pp. 42–48, 2007.

[10] K. Harada, T. Foissotte, T. Tsuji, K. Nagata, N. Yamanobe, A. Nakamura, and Y. Kawai. Pick and Place Planning for Dual-Arm Manipulators. In *Prceedings of 2012 IEEE Int. Conf. on Robotics and Automation*, pp. 2281–2286, 2012.

[11] J.-P. Saut, M. Gharbi, J. Cortés, D. Sidobre, and T. Siméon. Planning Pick and Place Tasks with Two-Hand Regrasping. In *Prceedings of 2010 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 4528–4533, 2010.

[12] M. Gupta and G.S. Sukhatme. Using Manipulation Primitives for Brick Sorting in Clutter. In *Proceedings of 2012 IEEE Int. Conf. on Robotics and Automation*, pp. 3883–3889, 2012.

[13] D. Katz, M. Kazemi, J.A. Bagnell, and A. Stentz. Clearing a Pile of Unknown Objects using Interactive Perception. In *Proceedings of 2013 IEEE Int. Conf. on Robotics and Automation*, pp. 154–161, 2013.

[14] R. Schnabel, R. Wahl, and R. Klein. Efficient RANSAC for Point-Cloud Shape Detection. In *Computer Graphics Forum*, pp. 214–226, 2007.

[15] S.A. Hutchinson and A.C. Kak. Planning Sensing Strategies in a Robot Work Cell with Multi-Sensor Capabilities. *IEEE Trans. on Robotics and Automat.*, Vol. 5, No. 6, pp. 765–783, 1989.

[16] J. Faigl, M. Kulich, and L. Preucil. A Sensor Placement Algorithm for a Mobile Robot Inspection Planning. *J. Intelligent Robot Systems*, Vol. 62, pp. 329–353, 2011.