Planning Dense Object Packing by Pushing Objects

Kazuki Iwao[1], Takao Nishi[1], Takuya Kiyokawa[1], Damien Petit[1], Weiwei Wan[1] and Kensuke Harada[1,2]

***Abstract*— To realize a dense object placement into a container, we propose a robotic packing motion planner by pushing objects to the side of other objects. Our method comprises three planning strategies, i.e., object placement planning, robotic packing-action planning, and action sequence planning. Object placement planning generates objects' placement into a container without gaps between objects. Based on the planned placement, the robotic packing-action planner selectively uses two action strategies where one is to directly place the object in the desired location of a container by using a pick-and-place approach, and the other is to first place the object at a certain distance from the surrounding object and then push it to achieve the placement without gaps. Finally, the action sequence planning plans the order of selected manipulation strategies. Through experiments, we confirmed that the robot efficiently packs multiple objects into a container by effectively using object pushing.**

I. Introduction

In recent years, with the rise of e-commerce, the expectation to fully automate the logistic processes is increasing. In particular, packing multiple types of goods into a single container (packing operation) is required when sending a required number of goods stored in a warehouse to retailers or consumers. However, this operation is still performed manually despite its heavy workload.

In order to automate the packing process with a robot, we have to solve two planning problems, i.e., one is to plan the object placement, where the location of objects placed in a container is planned, and the other is to plan the robot motion, where the object placement is realized with a robot. Object placement planning is known to be the knapsack problem of NP-hard. Approximate methods [1] and deep-learning-based method [2] have been proposed. Nevertheless, even if the object placement problem can be solved, it is not easy to realize the planned object arrangement by a robot, especially when a dense object arrangement is planned. To achieve such an object arrangement, a robot must insert the grasped object into a narrow gap between two surrounding objects or beside a container's surface. The task is challenging since even with a small position error of the inserted object, the insertion may fail due to the collision between objects or between the robot's gripper and an object.

To cope with such a problem, we propose a method for robotic motion planning for packing objects into a container. Our method can realize the dense object placement with no gap between objects. Instead of directly placing objects in the target place, our method first places the grasped object at a certain distance from other objects which have already been placed in a box. Then the robot pushes the target object to the side of another object to realize the object placement without gaps as shown in Fig.1. During the process, the motion planning has to be performed under some constraints. First, the area on the bottom surface of a container used for this motion planning should be as small as possible. Second, the number of motion sequences used to realize the dense object placement should be as small as possible. For this purpose, our motion planning problem includes the sequence planning problem which plans the order of objects to be pushed.

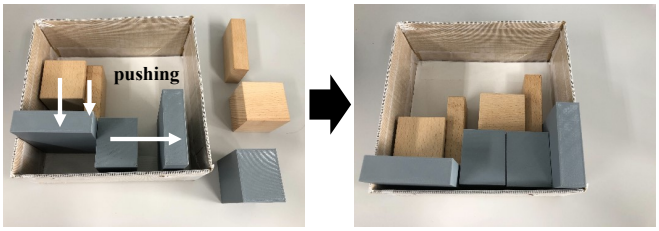


Fig. 1. Dense object packing into a container by pushing an object to the side of another object.

Specifically, we first determine the placement of each object on the bottom surface of a container by extending the Bottom-Left (BL) method [1] assuming the arbitrary order of objects to place. Then, we select the order of objects to place occupying the minimum area on the bottom surface of the container. To plan the object-packing action, one of the following three actions is selected to place each object: (1) directly placing the object in the target position, (2) placing the object at a certain distance from other objects and then pushing it to realize the object placement without gap, and (3) placing multiple objects at a certain distance from others and then simultaneously pushing multiple objects to realize the dense object placement. A set of actions where the robot realizes the object placement which consumes the minimum area on the bottom surface of the container with minimum number of action sequences without causing the collision with the container wall is selected. Finally, among the possible sequence of actions, the action sequence which avoids collisions between objects is determined.

We performed real experiments by using a robotic manipulator with two-fingered gripper attached at the tip. Through experiments, we confirmed that using our method the robot realizes dense object placement with no gap between objects. In addition, by utilizing the object pushing, the length of action sequence could be effectively reduced. Furthermore,

*This work was not supported by any organization

[1] Graduate School of Engineering Science, Osaka University, Japan, iwao@hlab.sys.es.osaka-u.ac.jp

[2]National Institute of Advanced Industrial Science and Technology (AIST), Japan, harada@hlab.sys.es.osaka-u.ac.jp

our method can reduce the area on the bottom surface used by the pushing operation.

The remainder of the paper is organized as follows: we introduce the related works in Section II and present the proposed method in Section III. Experiments are conducted in Section IV. The results are discussed in Section IV-C. Finally, we present our conclusions and future perspectives in Section V.

II. Related Works

In this section, we review previous works on object packing problem planning and robotic motion planning related to object packing.

A. Object packing

The object packing problem can be classified into two categories, i.e., online and offline ones. Online packing is the problem of sorting all objects according to various rules and determining the position of each object in a box in the sorted order. Since the packing problems are known to be NP-hard, approximate solutions have been proposed. For example, the Bottom-Left (BL) method for 2D or Deepest-Bottom-Left (DBL) method for 3D bin-packing problem with the computational complexity of O($N^3 \log N$) have been proposed [1]. Richa et al. [3] improved the computing speed of the bin-packing problem by using deep reinforcement learning. Zhao et al. [2] used deep learning to place the object in a position where the next object is easy to place. Hauser et al. [4] achieved a robotic packing method that considers the stability of each object based on the BL method.

In contrast to online packing, offline packing incrementally improves the quality of the solution by changing the object position. Tanaka et al. [5] converted the packing problem into a linear programming problem and applied the greedy method after relaxing the constraints to obtain the optimal solution. Kobatake et al. [6] reduced the computation time of a 3D packing problem by representing the positions of all the objects in a 3D slice tree. The genetic algorithm has been used for 1D [7] and 3D [8]–[10] offline bin-packing problems. Hauser et al. [11] proposed a method for inserting irregular 3D objects into containers, independent of the arrival order. Yasuda et al. [12] applied the particle swarm optimization for offline packing.

However, the conventional methods on the object packing problem do not consider the feasibility of robot motion. When the filling rate is increased, a collision may occur with the gripper, container wall, and/or objects. In this regard, we propose a novel approach of robotic packing-motion planning by utilizing object pushing.

B. Manipulation planning

There have been a number of works on task and motion planning for robotic manipulation [13], [14]. For retrieving an object from clutter, there have been a number of robotic manipulation planning approaches, such as relocating surrounding objects [15]–[18] and using bimanual manipulation [19], [20]. However, there has been no research on robotic manipulation planning for dense object packing into a container by utilizing object pushing.

III. Proposed Method

A. Definition and method overview

Our method is developed under the assumption where many rectangular objects of several sizes are placed on a table and the position and orientation of each object are known. Fig.2 shows the overview of our proposed method. It comprises three planning strategies, i.e., object placement planning, robotic packing-action planning, and action sequence planning. We first carry out the object placement planning to plan the pose of objects densely placed in a container. Next, we plan the robotic packing actions by assigning one of the two placing methods to each object, i.e., the direct and indirect placing methods. The direct placing method directly places the object to the target position by pick-and-place. On the other hand, the indirect placing method once places the object at a certain distance from other objects. According to the number of objects pushed simultaneously, we can further classify the indirect placing into two methods, i.e., (1) placing the object at a certain distance from other objects and then pushing it to reach the target position, and (2) placing multiple objects at a certain distance from others and then simultaneously pushing multiple objects to reach the target position. When indirect placing is used, the number of action sequences increases significantly if only one object is pushed at each sequence, which worsens the task efficiency. Therefore, in our packing-action planning, we first try to push multiple objects simultaneously in the same direction. The object placing method is determined for each object from direct placing, indirect placing for a single object, and indirect placing for multiple objects, to minimize the length of action sequences and to avoid collisions between gripper and container. In addition, our action planning works to reduce the area on the bottom surface of the container consumed by the object pushing. Furthermore, The action sequence planning plans the order of objects to place that minimizes the travel distance of the gripper.

B. Object placement planning

Our method on object placement planning is a simple and heuristic one to provide an object placement near maximum density without gaps between two objects. It extends the BL method by assuming arbitrary order of objects to realize the dense object placement. In addition, the dense object placement can also be realized by maximizing the contact area between two objects and between an object and the container wall.

Let us consider N objects placed on the table where each object is named as $O_1, O_2, \cdots, O_N$ according to the order of objects defined in the BL method where the different name is assigned to each object if the order of objects to place changes. Σ_C denotes the coordinate system attached at the front-left corner of the container's bottom surface where z axis is perpendicular to the bottom surface. Fig.3 shows the overview of the object placement planning method. First, we

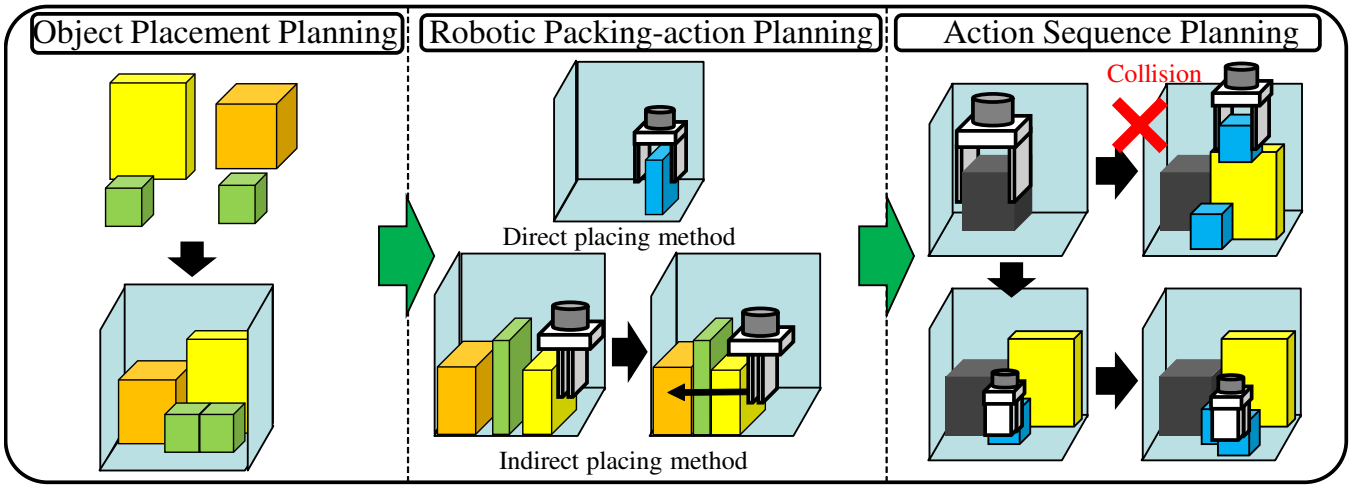


Fig. 2. Overview of the proposed packing method.

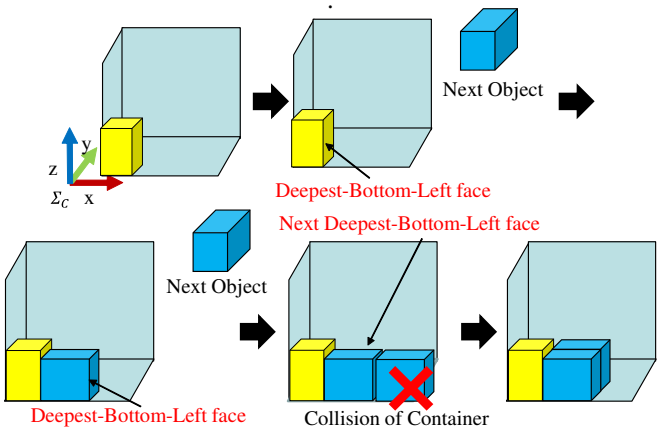


Fig. 3. Overview of object placement planning.

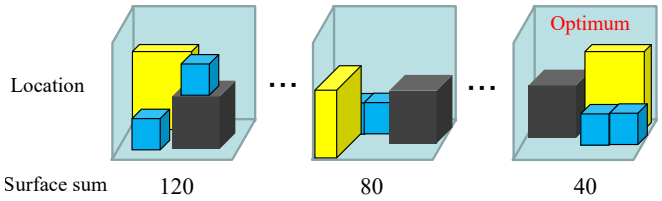


Fig. 4. Improvement of object density placed in a container.

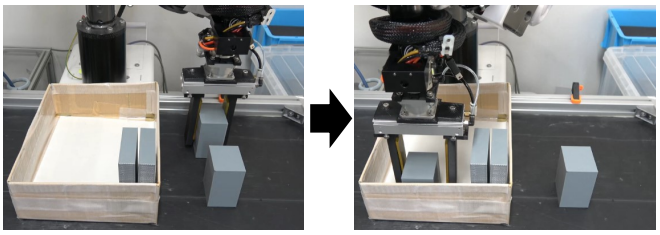


Fig. 5. Direct placing method. (Pick-and-place the object to target position)

find the surface $S_{1,1}$ of O_1 with the largest area. We place O_1 in the container such that $S_{1,1}$ contacts the $y-z$ plane of the container wall. Next, the object O_2 is placed next to O_1 based on the BL method such that the surface $S_{2,1}$ of O_2 where the contact area between two O_1 and O_2 is maximized contacts the surface of O_1 on the opposite side of $S_{1,1}$. The above procedure is repeated until all the objects are placed in the container, and determine the placement pose of all objects. Furthermore, we iterate this object placement process by randomly changing the order of objects. Finally, we select the object placement with minimum area of objects occupied on the bottom surface of the container. This object placement planning is an extension of the BL method where we added the determination of the object orientation and order of objects to the original BL method.

In this way, we construct an object placement without gap between two objects. In our method, the contact area between two adjacent objects is maximized since it likely generates the dense object placement with minimum area of objects occupied in the bottom surface of the container as shown in Fig.4.

C. Robotic packing-action planning

The following section introduces robotic packing-action planning by utilizing direct and indirect placing methods. Overview of the direct and indirect placing methods are shown in Figs.5 and 6, respectively. In indirect placing, objects are placed at distance d in x or y direction from the neighboring object and then pushed to the target position. Here, the minimum value of d is the thickness of the finger. We call the pushing in the x and y directions as the row pushing (Fig.7) and the column pushing (Fig.8), respectively.

We first explained how to choose a placing method in III-C.1. Then, our indirect placing method utilizes the object pushing, an objects-pushing method including the simultaneous pushing of multiple objects is introduced in III-C.2.

In these subsections, we assign a placing action to each object and do not determine the action order. The action sequence is planned in the next subsection by constructing the action graph.

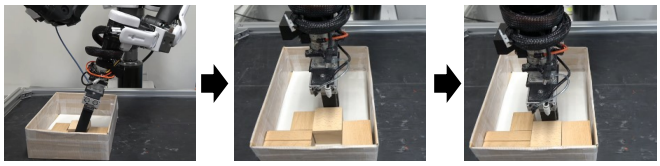


Fig. 6. Indirect placing method. (Pick-and-place and push to target position)

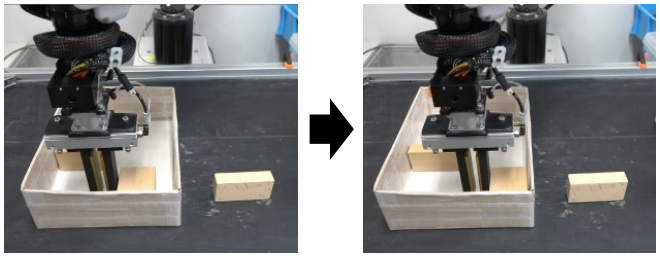


Fig. 7. Row pushing.

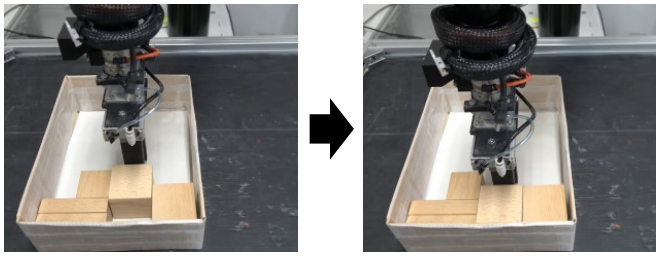


Fig. 8. Column pushing.

1) Choosing packing actions: We first explain the method for assigning an action to each object. The action of each object is planned such that the collision between two objects and between the robot and surrounding objects is avoided.

The pseudo-code of the packing-action planning is shown in algorithm 1. In this algorithm, we define the inputs as $objPose_{list}$: the list of objects' placement pose, $objName_{list}$: the list of objects' names (number), and the outputs as $minSeqSize$: the minimum size of action sequences, $endNode$: the solution node of the action graph obtained by Algorithm 2. In addition, we use the variables $Direct_{list}$: the list of objects names where the direct placing is assigned and $Indirect_{list}$: the list of objects where the indirect placing is assigned. The node of the action graph defined in Algorithm 2 includes $cost$: travel distance of the gripper from the root to the node, $depth$: depth of the node, Act_{order}: order of actions. Using these variables, the function planPushDir outputs $PushDir_{list}$: the list of pushing direction of each object and $Area$: area used in the direct and indirect placing actions at the bottom surface of the container. If the pushing directions of adjacent objects are the same, we check whether the multiple objects can be simultaneously pushed by using the method shown in subsection III-C.2. If Col: a boolean value determining if the collision occurs is true while pushing, the pushing action is judged as failure. Otherwise, we use the function countSeqSize to obtain $SeqSize$: the sum of the total number of placing actions,

Algorithm 1 Robotic Packing-Action Planning

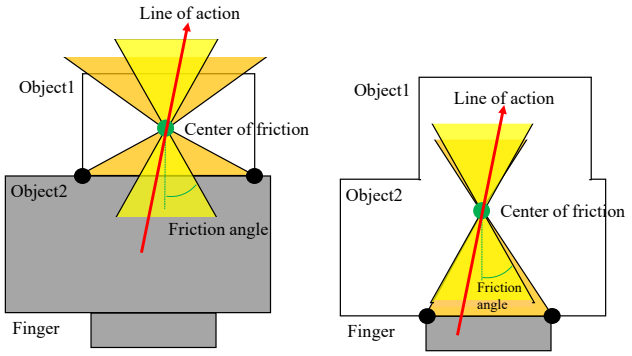
Input: $objPose_{list}$: List of objects' placing poses, $objName_{list}$: List of object names(numbers)
Output: $minSeqSize, endNode$
Struct $Node$
  $cost$
  $depth$
  Act_{order}
end
for $i \leftarrow 0$ to N **do**
  $Direct_{list} \leftarrow {}_N\mathrm{C}_i$
  for $j \leftarrow 1$ to ${}_N\mathrm{C}_i$ **do**
    $Indirect_{list} \leftarrow objName_{list} - Direct_{list}$
    for $k \leftarrow 1$ to 2^{N-i} **do**
      $(PushDir_{list}, Area) \leftarrow$ planPushDir($objPose_{list}, Indierct_{list}, Direct_{list}$)
      if Col **then**
        continue
      end if
      $SeqSize \leftarrow$ countSeqSize($Indirect_{list}, Direct_{list}, objPose_t, PushDir_{list}$)
      $(Act_{list}, ActCost_{list}) \leftarrow$ createActionList($Indirect_{list}, objPose_{list}, Direct_{list}, PushDir_{list}$)
      if $(SeqSize > minSeqSize) \mid ((SeqSize = minSeqSize)\ \&\ (Area > minArea))$ **then**
        continue
      end if
      $minSeqSize \leftarrow SeqSize$
      $minArea \leftarrow Area$
      planActSeq($curNode, Act_{list}, ActCost_{list}, minSeqSize, minNode$)
      if $minNode.cost < endNode.cost$ **then**
        $endNode \leftarrow minNode$
      end if
    end for
  end for
end for

and calculate the number of indirect placing actions. Then, we create Act_{list}: the list of actions and $ActCost_{list}$: the list of costs associated with the actions based on $Indirect_{list}$, $objPose_{list}$, $Direct_{list}$, and $PushDir_{list}$.

If $SeqSize$ is larger than the current minimum, another placing action is assumed. If $SeqSize$ is equal to $minSeqSize$ and if $Area$ is larger than its current minimum, another placing action is assumed. Otherwise, substitute $SeqSize$ to $minSeqSize$ and $Area$ to $minArea$. Then, the function $planActSeq$ is executed to determine the order of actions associated with object placement. The function $planActSeq$ will be explained in details in the algorithm 2, where $minNode.cost$: the node with minimum cost is obtained. If $minNode.cost$ is smaller than the preserved

one, information on the preserved Node is replaced by the information on the minNode.

2) Pushing multiple objects: In this research, a robot may simultaneously push multiple objects to reduce the length of action sequences. We use a method introduced for simultaneously pushing multiple objects [21] . The overview of the method is shown in Fig.9. A robot pushes an object by closing the fingers and using the finger's back surface. When pushing a single object by using the translational motion of the finger, relative motion between the object and finger does not occur if the contact force is included strictly inside the friction cone and if the line of action passes strictly inside the contact segment. When simultaneously pushing two objects, we consider the situation where the finger contacts object 2, and object 2 contacts object 1. Consider the contact between two objects. The relative motion between two objects does not occur if the contact force is included strictly inside the friction cone of the contact and if the line of action of the contact passes strictly inside the contact segment. Similarly, consider the contact between object 1 and the finger. The relative motion between object 1 and the finger does not occur if the contact force is included strictly inside the friction cone of the contact and if the line of action of the contact and the finger passes strictly inside the contact segment. Here, under the quasi-static assumption, the contact force can be obtained by integrating the normalized velocity multiplied by the friction coefficient over the contact area. In addition, the line of action passes through the center of friction. For simplicity, we assume the uniform friction distribution with known friction coefficient. In this case, the center of friction coincides with the geometrical center.



(a) Condition for preventing relative motion between object 1 and object 2

(b) Condition for preventing relative motion between object 2 and finger

Fig. 9. Mechanical relation of multiple objects pushing.

D. Action sequence planning

In this section, we present the action sequence planning by using the packing action planned for each object. We search for the minimum number of action sequences without making collisions between the container wall, the object, and between objects. Here, the order of action is determined by constructing the tree structure [22] and searching for it.

Algorithm 2 Action Sequence Planning

Input: $curNode$: current node, Act_{list}: list of action assigned to each object, $ActCost_{list}$: list of action cost, $leafDepth$: depth of leaf node, $minNode$: node with minimum cost

```
if curNode.depth = leafDepth then
  if curNode.cost < minNode.cost then
    minNode ← curNode
  end if
  return
end if
if curNode.cost > minNode.cost then
  return
end if
for i ← 0 to N do
  for k ← 0 to curNode.depth do
    if Col then
      continue
    end if
  end for
  curNode.Act_{order,k} ← Act_{list,i}
  curNode.cost ← curNode.cost + ActCost_{list,i}
  curNode.depth + +
  planActSeq(curNode, ActCost_list, Act_list,
        leafDepth, minNode)
  delete(curNode.Act_{order,k})
  curNode.cost ← curNode.cost − ActCost_{list,i}
  curNode.depth − −
end for
```

The details of the action sequence planning algorithm 2 are described in the following where its maximum computational complexity is $O(N!)$.

The algorithm is a recursive structured method that implements a tree-structured action graph where each node stores the total cost from the root, the depth, and the order of action from the Root. The weight of an edge is the action cost where the action cost is the total travel distance of the gripper from the hand's current position to the placement position of the object.

We define the inputs as $curNode$: current node, $leafDepth$: depth to the leaf node, $ActCost_{list}$: the list of action costs, $minNode$: the node with minimum cost. If the current node is the leaf node and the travel distance of the gripper is smaller than the cost of the current node, $minNode$ is replaced by the current node. On the other hand, if the cost of the current node is larger than $minNode.cost$, the algorithm terminates after pruning branches to reduce the number of searches. Next, for the action $Act_{list,i}$ is applied to the k-th element of the list $curNode.Act_{order}$. If there is no collision, the information on the current node is updated and recursively executes algorithm 2 again. In this way, we search for the action graph until the last leaf node. Using this tree structure, it is possible to determine the order of action

with the computational cost less than $O(N!)$, since branch pruning is performed in the case of collisions or when the tentative minimum cost is exceeded.

IV. Experiment

In this section, we carry out two experiments to confirm the effectiveness of the proposed method. In the first one, two kinds of four objects are to be placed in a container. In the second one, four kinds of eight objects are to be placed in a container.

A. Experimental setup

The three planning problems including object placement planning, robotic packing-action planning and action sequence planning are implemented on the software platform Choreonoid [23]. We used the robot named Nextage developed by Kawada Robotics Inc. for the experiment, where a two-fingered gripper is attached at each arm tip. The robot pushes the object by using the back surface of the closed finger.

B. Experiment results

In the first experiment, the proposed method was applied for packing two types of four rectangular objects into a container. The size and initial state of each object are shown in Fig.10 and these objects are made by 3D printer. Objects with the same number have the same size.

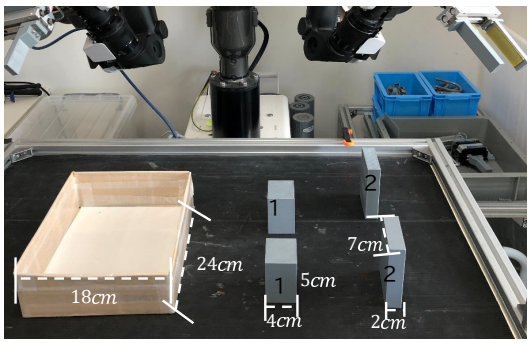


Fig. 10. Initial state of experiment on packing two kinds of four objects.

The total number of action sequences was calculated as 6: one simultaneous pushing of two objects, one pushing, three indirect placings, and one direct placing. The sequence of motions was: object2 (indirect placing) → object2 (indirect placing) → object2 and object2 (simultanelous pushing) → object1 (direct placing) → object1 (indirect placing) → object1 (pushing). Fig.11 shows the obtained action sequence.

Next, we performed an experiment of packing four types of eight rectangular objects into a container. The size and initial state of each object is shown in Fig.12. The objects 1 and 4 are made by 3D printer while the objects 2 and 3 are made in wood. Objects with the same number have the same size.

In this experiment, the total number of action sequence was calculated as 13: two simultaneous pushing, three pushing, seven indirect placings, and one direct placing. The

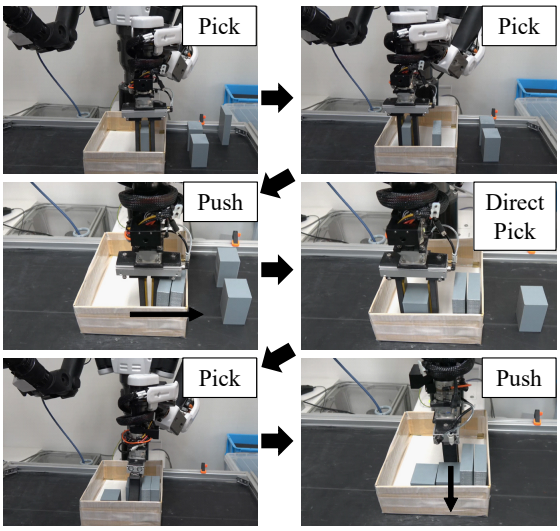


Fig. 11. Experimental results of packing two kinds of four objects.

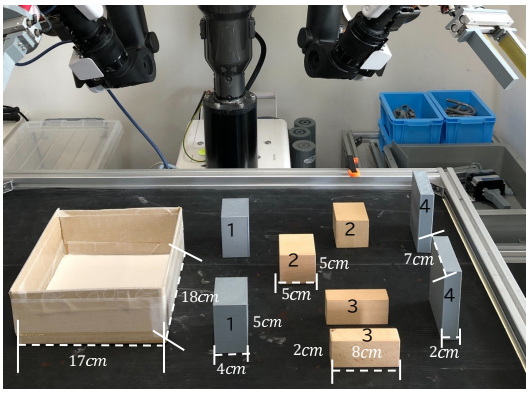


Fig. 12. Initial state of experiment on packing four kinds of eight objects.

sequence of motions was: object4 (indirect placing) → object1(indirect placing) →object1 and object4 (simultanelous pushing) → object4 (indirect placing) → object4 (pushing) → object2 (direct placing) → object3 (indirect placing) → object2 (indirect placing) → object3 (indirect placing) → object4 (indirect placing) → object2 and object4 (simultanelous pushing) → object3 (pushing) → object3 (pushing). The sequence of actions of this experiment is shown in Fig.13.

C. Discussion

We first discuss how densely the objects can be placed in our proposed method. The experimental result of packing two kinds of four objects is compared to our previous work [24] as shown in Table I where it showed better performance by providing a reduced area on the bottom surface of the container occupied by objects. The method used in our previous work [24] does not consider multiple placement states while our method does.

Next, the length of the action sequence for the experiment of four kinds of eight objects is shown in table II. The length of the action sequence was 13 when using the proposed method, while the length was 14 without using direct placing, in addition, the length was 15 without

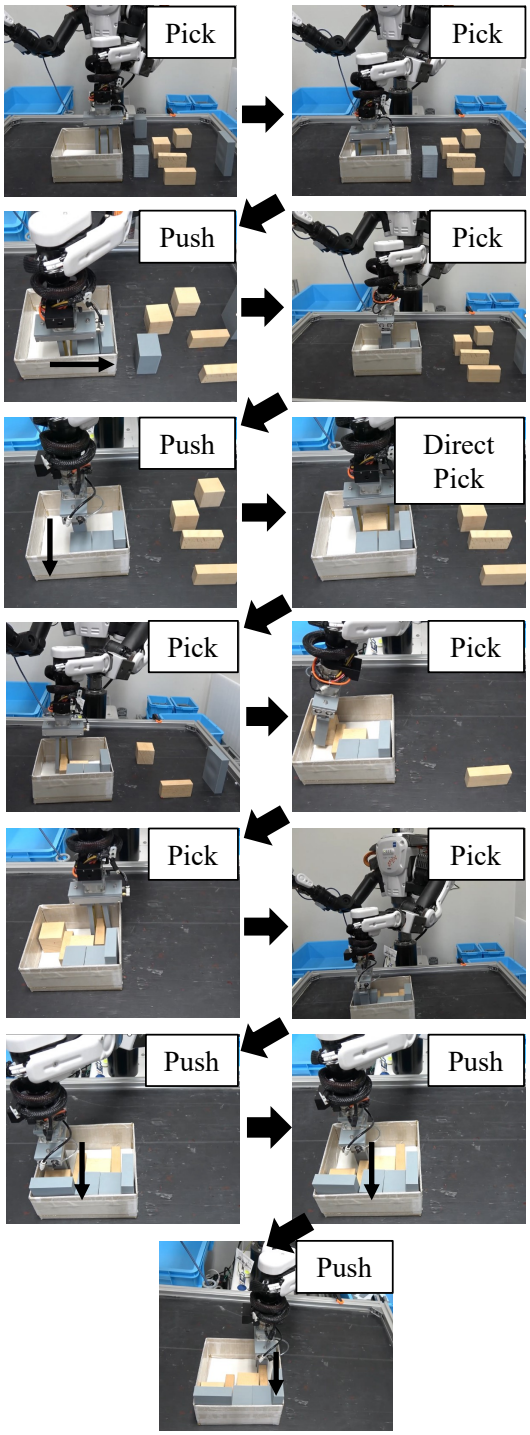


Fig. 13. Experimental results of packing four kinds of eight objects.

TABLE I

OBJECT PLACEMENT IN PACKING TWO KINDS OF FOUR OBJECTS

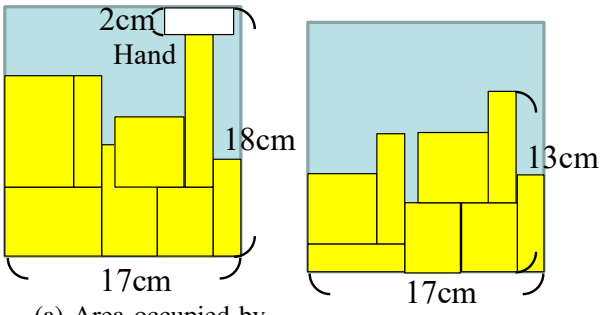
Placement method	$surfacesum(\mathrm{cm}^2)$
Proposed	138
Previous [24]	210

using simultaneous multiple-object pushing. Furthermore, the length of action sequence using neither direct placing nor simultaneous multiple-object pushing was 16. From these results, we can see that the length of action sequence was successfully reduced by using both direct placing and simultaneous multiple-object pushing. Thus, the proposed method successfully determines the action sequence which can be obtained without causing collision.

TABLE II

COMPARED WITH THE LENGTH OF ACTION SEQUENCE IN PACKING FOUR KINDS OF EIGHT OBJECTS

Motion method	Motion count
Ours	13
w/o direct placing	14
w/o pushing multiple objects	15
w/o direct placing and pushing multiple objects	16



(a) Area occupied by objects during the packing action.

(b) Area where the objects are finally placed.

Fig. 14. Area of placed objects in the experiment of packing four kinds of eight objects.

Fig.14 (a) shows the area on the bottom surface of a container occupied by objects during the packing action. On the other hand, Fig.14 (b) shows the area where objects were finally placed. These results show that our proposed method can provide minimum area on the bottom surface of the container used by object pushing action. This contributes to reducing the size of containers used for packing objects.

V. CONCLUSION

In this paper, we proposed a robotic packing motion planner by pushing objects to the side of other objects to realize a dense object placement into a container. Our method selectively uses two action strategies, i.e., direct placing of objects in the desired location and indirect placing where the objects are placed at a certain distance from the surrounding object and then pushed to achieve the placement without gaps. From experimental results, we confirmed that, by using our proposed method, a robot could realize a dense object placement placed into a container compared to our previous method. In addition, by using both the direct placing and multiple-object pushing, the length of action sequence could be effectively reduced. Furthermore, our method can reduce the area on the bottom surface used by the pushing operation.

In the future, we plan to ensure enough gap between two objects when the object is first placed. In addition, we plan to implement a system that recognizes the position of objects. Furthermore, we plan to extend the proposed method

to create a placement state for each object for an irregularly shaped object and to realize automatic box-packing operation using various robotic motion plans, such as simultaneous pushing.

REFERENCES

[1] Y. Tanaka, H. Kawashima, S. Imahori and M. Yanaura, "An efficient implementation of a constructive algorithm for the three-dimensional packing problem" in AL, pp. 1–8, 2013.

[2] H. Zhao, C. Zhu, X. Xu, H. Huang and K. Xu, "Learning Practically Feasible Policies for Online 3D Bin Packing", arXiv preprint arXiv:2108.13680, 2021.

[3] V.Richa et al,. "A Generalized Reinforcement Learning Algorithm for Online 3D Bin-Packing", arXiv preprint arXiv:2007.00463, 2020.

[4] F.Wang and K.Hauser, "Stable bin packing of non-convex 3D objects with a robot manipulator", *IEEE Int. Conf. on Robotics and Automation*, 2019.

[5] Yuma Tanaka and Mihiro Sasaki and Mutsunori Yagiura, "A Heuristic Algorithm for the Node Capacitated In-tree Packing Problem", Journal of Information Processing(JIP), 2009.

[6] N. Kobatake, H. Ohta, and M. Nakamori, "A Study on Three Dimensional Packing Problem Considering the Order of Loading and Unloading", IPSJ SIG Technical Report, 2011.

[7] T. Yakawa and N. Sannomiya, "Proposition of Genetic Algorithm for Bin Packing Problems", Trans. of the Society of Instrument and Control Engineers, 2005.

[8] Y. Wu, W. Li, M. Goh, and R. de Souza, "Three-dimensional bin packing problem with variable bin height", European J. Operational Research, 2010.

[9] V.A. Chekanin and A.V. Chekanin, "Multimethod genetic algorithm for the three-dimensional orthogonal packing problem", J. Physics: Conference Series, 2019.

[10] R. Sridhar, M. Chandrasekaran, C. Sriramya and T. Page, "Optimization of heterogeneous Bin packing using adaptive genetic algorithm", *IOP Conf. Series: Materials Science and Engineering*, 2017.

[11] F. Wang and K. Hauser "Robot packing with known items and nondeterministic arrival order", IEEE Trans. on Automation Science and Engineering, 2020.

[12] A. Yasuda, G.A. Garcia-Richardez, J. Takamatsu and T. Ogasawara "Packing Planning and Execution Considering Arrangement Rules", *IEEE Int. Conf. on Robotic Computing (IRC)*, 2020.

[13] J. Wolfe, B. Marthi, and S. Russell, "Combined task and motion planning for mobile manipulation," *Int. Conf. on Automated Planning and Scheduling*, 2010.

[14] C. Garrett, T. Lozano-P´erez, and L. Kaelbling, "Ffrob: an efficient heuristic for task and motion planning," *Algorithmic Foundations of Robotics XI*, pp. 179–195, 2015.

[15] H. Song, J.A. Haustein, W. Yuan, K. Hang, M. Y. Wang, D. Kragic and J. A. Stork, "Multi-object rearrangement with monte carlo tree search: A case study on planar nonprehensile sorting", *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2020.

[16] C. Song and A. Boularias, "Object rearrangement with nested nonprehensile manipulation actions", *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2019.

[17] L. Chang, J.R. Smith R and D. Fox, "Interactive singulation of objects from a pile", *IEEE Int. Conf. on Robotics and Automation*, 2012.

[18] E. Huang, Z. Jia and M. T. Mason, "Large-scale multi-object rearrangement", *IEEE Int. Conf. on Robotics and Automation*, 2019.

[19] T. Motoda, D. Petit, W. Wan, and K. Harada, "Bimanual shelf picking planner based on collapse prediction," *IEEE Int. Conf. on Automation Science and Engineering*, pp. 510—515, 2021.

[20] S. Nagato, T. Motoda, K. Koyama, W. Wan and K. Harada, "Motion Planning to Retrieve an Object from Random Pile", Int. Conf. on Artificial Life and Robots, 2022.

[21] K. Harada, J. Nishiyama, Y. Murakami and M. Kaneko "Pushing multiple objects using equivalent friction center", *IEEE Int. Conf. on Robotics and Automation*, 2002.

[22] T. Sakamoto, W. Wan, T. Nishi, K. Harada "Efficient Picking by Considering simultaneously Two-Object Grasping", *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2021.

[23] T. Tsuji and K. Harada "Graspplugin for Choreonoid", J. Robotics Society of Japan, 2013.

[24] K. Iwao, K. Koyama, W. Wan, T. Nishi and K. Harada "Grasp/Motion Planning for Efficient Packing Tasks Assuming Multiple Grippers", *JSME Annual Conf. on Robotics and Mechatronics*, 2021.