Planning for grasping cluttered objects based on obstruction degree

Wenrui Zhao¹, Jingchuan Wang¹, Weidong Chen¹ and Yi Huang²

Abstract
Grasping objects in clutter is more difficult than grasping a separated single object. An important issue is that unsafe grasps may occur, in case, one object sits or leans on another, which could cause the collapse of objects. In addition, reachability of each object surrounded by other obstacles also has to be considered. So the order of multiple objects for grasping and the grasp configuration of each object must be planned simultaneously. This article combines grasp order and grasp configuration planning to perform fast and safe multiobject grasping in cluttered scenes. First, a comprehensive grasp configuration database is built to provide enough feasible grasp configurations for the objects. Then, we propose an obstruction degree to estimate the likelihood of reachability of each grasp configuration as well as each object. This measurement also implicitly infers object interactions. Finally, grasp order and grasp configurations are planned together to deal with the constraints caused by reachability and object interaction. Simulations and experiments in a series of cluttered scenes demonstrate that our method can grasp objects efficiently and can greatly reduce unsafe grasps.

Keywords
Grasp planning, grasp order, grasp configuration, clutter, object interaction

Date received: 12 January 2021; accepted: 02 August 2021

Introduction
Multiobject grasp planning in cluttered scenes has been regarded as a key problem in robot manipulation field. Grasping multiple objects is constrained by reachability and object interaction. On the one hand, there are various grasp configurations for different objects, and each object is surrounded by other objects, so one reachable grasp configuration must be chosen for each object. On the other hand, objects will collapse if an object that is supporting other objects is removed first, and objects may be damaged and may roll to unreachable places. Therefore, an appropriate grasp order and reachable grasp configurations for all objects are required to achieve safe and fast grasping in cluttered scenes.

Many researchers have solved grasping known objects by dividing the whole process into two steps: offline grasp generation and online grasp planning. Shape primitive¹–³ is widely used in grasp generation. It approximates the target object to one or more shape primitives, which include an appropriate set of grasp configurations. Region masking⁴–⁶ is also a popular methodology. It can quickly locate the region, which has a high probability of containing high-quality grasp configurations. Tsuji et al.⁷ combine the above two methods, select the constricted region of the

¹ Department of Automation and Institute of Medical Robotics, Shanghai Jiao Tong University, and Key Laboratory of System Control and Information Processing, Ministry of Education, Shanghai, China
² Nanjing Kingyoung Intelligent Technology CO., LTD.

Corresponding author:
Jingchuan Wang, Shanghai Jiao Tong University, No. 800, Dongchuan Road, Shanghai 200240, China.
Email: jchwang@sjtu.edu.cn
object as the grasp interest region, and fit the local model of the object near the grasp interest region to the grasp primitives. Li et al.⁸ wrap ropes around the object to find possible grasp regions and then computes the contacts of a multi-fingered hand with object surfaces around these regions. Wan et al.⁹ apply superimposed segmentation to the object mesh model and use the uniform facets to locate contacts and generate grasp poses for grippers and suction cups. Grasp generation only provides possible grasp configurations but does not decide the final grasp configuration to grasp an object in a cluttered scene.

Online grasp planning is usually used to decide how to grasp each object in clutter. One type is planning on action level, which finds the best way to grasp one object in the current scene. Most studies usually choose some metric as objective function to optimize grasp configurations in the scene. Berenson et al.¹⁰ propose a grasp objective function that takes into account the kinematics of the robot, the environment around the object, and the grasp force-closure quality. Some studies are interested in planning a path that approaches the target while pushes away other obstacles¹¹⁻¹³ or pushing the target itself to make it graspable.¹⁴,¹⁵ The other type is planning on task level, which finds an order of objects to grasp in the current scene. Stilman et al.¹⁶ and Dogar and Srinivasa¹⁷ consider occlusion between objects and plan how to move the objects to get a target object. But they cannot deal with complex object interactions like stacking or slanting. Some studies work on recognizing complex object interactions,¹⁸⁻²² which is used to decide a safe grasping order. Due to neglect of concrete grasp configuration planning, the actual feasibility of the plan cannot be guaranteed.

In addition, a lot of research also solves grasping unknown objects, which usually generates grasp configurations online. Lippiello et al.²³ reconstruct object surface through images taken by a camera and move the fingertips on the surface to find optimal grasp. Lei and Wisse²⁴ extract object’s concave hull contour from the point cloud and calculate suitable grasping by maximizing the coefficient of force balance. Lin et al.²⁵ transfer example grasps taught by human demonstration to similar objects. Now deep learning²⁶⁻²⁸ is becoming the most prevailing approach to solve grasping problems. However, these methods usually consider scenes where objects are sparsely placed or object collapse is not detrimental.

In summary, most of the current research on multiobject grasp planning separates task level and action level or only involves one of the two levels. We aim at combining the two planning levels. Concretely, we focus on a task that a robotic arm with a two-finger parallel gripper is commanded to grasp all objects on a table. The order of objects to grasp and reachable grasp configuration for each object are planned simultaneously, for the purpose that the risk of object collapse is reduced and reachable grasp configurations are found quickly. We hold several assumptions:

- Object three-dimensional (3D) models are known.
- Identifications (IDs) and poses of visible objects can be recognized.
- No object belongs to containers, that is, an object is never in another object.

Grasp planning based on obstruction degree (OD) is proposed to realize safe grasp order and fast grasp configuration search. Differently from previous works, we do not infer object interactions explicitly using complex algorithms. Our method is very simple and easy for implementation and has high efficiency in planning.

The structure of this article is as follows: The second section gives a glimpse of the overall planning framework; the third section describes the method used to generate grasp configurations offline; the fourth section proposes OD computation and online planning algorithms; the fifth section evaluates our planning framework by simulations and experiments; and the sixth section gives a conclusion and discusses the future work.

**Figure 1. Proposed grasp planning framework.**

1. Object three-dimensional (3D) models are known.
2. Identifications (IDs) and poses of visible objects can be recognized.
3. No object belongs to containers, that is, an object is never in another object.

**Grasp planning framework**

The whole planning framework is shown in Figure 1. Grasp generation only needs to be computed once offline and provides a grasp database with widely distributed grasp center points and grasp approach directions. Grasp
planning is performed online to decide grasp order and grasp configurations for all objects based on ODs.

**Offline grasp generation**

Region masking extracts the object mean curvature skeleton $SK$ and samples interested vertices according to the connectivity to their neighbors. Then hypotheses generation applies to search in a spherical grid built on each interested vertex to find collision-free grasp hypotheses $H$. Next, grasp stability analysis formulates a grasp quality function to filter unstable grasps from the generated hypotheses and the remains are stored in the grasp database $B$.

**Online grasp planning**

Grasp candidates selection selects possible grasp candidates $B_0$ that may be reached by the robot in the current scene from the database $B$. Then, obstruction analysis determines ODs between grasp configurations and objects as well as between objects and objects in the current scene. Next, grasp order planning plans the order $X_n$ of all objects to grasp, which minimizes total ODs. At last, grasp configuration planning searches reachable grasp configurations $Y_n$ for all objects.

**Grasp generation**

**Grasp parameterization**

A grasp is commonly parameterized by gripper-specific information and gripper pose. Following parameters define a grasp configuration $P$ for a two-finger parallel gripper:

- $P_c$: grasp center point: a point at which the center between the two fingertips is located.
- $P_d$: grasp approach direction: a vector that indicates the direction the fingers point to.
- $P_r$: gripper roll angle: the roll angle of gripper around the approach direction.
- $P_s$: opening degree: the distance between the two parallel fingers before grasping.

$P_c$, $P_d$, and $P_r$ are described in object coordinate frame and determine the gripper pose when grasping an object. The gripper usually first moves to a pregrasp pose and then moves forward along $P_d$ to arrive at the final grasp pose. This is why $P_d$ is called “grasp approach direction.” As a grasp configuration can be determined by the above parameters, grasp generation can be regarded as a sampling process for these parameters.

**Region masking**

We assume that 3D models of the objects are available. To reduce computation effort of grasp generation and online planning, the models can be simplified, for example, as unions of primitive shapes.1–3

According to human grasp experience, the surrounding areas of skeleton vertices are usually used as preferred areas for humans to grasp objects.29 We choose the mean curvature skeleton30 as the interested region, like Vahrenkamp et al.6 have done. A resulting skeleton is a graph $SK = (V, E)$, in which each vertex $v \in V$ is connected to one or multiple neighbors via edges $e \in E$. According to the connectivity to neighboring vertices, each skeleton vertex $v$ can be classified into branching vertex, endpoint vertex, and connecting vertex. We select all branching and endpoint vertices as interested vertices, and connecting vertices are uniformly sampled as interested vertices.

**Hypotheses generation**

Grasp parameters are sampled according to Algorithm 1. First, a spherical coordinate frame is established at an interested vertex (line 2), and the $x, y$, and $z$ axes are parallel with that of the object coordinate frame as depicted in Figure 2. Then point $(r, \theta, \phi)$ inside the object model is sampled (lines 3, 4, and 6) as the grasp center point $P_c$. The grasp approach direction $P_d$ is parallel with the direction $(\theta, \phi)$ and points to the interested vertex; gripper roll angle $P_r$ is also sampled (line 7). $P_s$ determines the gripper position, $P_d$ and $P_r$ determine the gripper rotation, and $P_s$ is conservatively predefined as the largest opening degree of the gripper. At this time, a grasp hypothesis can be generated (line 8). If the gripper in that configuration is not in collision with the object model (line 9), the hypothesis will be inserted into the set of grasp hypotheses $H$ (line 10).

A grasp hypothesis for a two-finger parallel gripper can also be seen as $h = (P_h, P_s)$. $P_h$ is the pose of the gripper in object coordinate frame and can be computed by

| Algorithm 1. Search in Spherical Grid. |
|----------------------------------------|
| **Input:** object model $M$, interested vertex $v$, opening degree $P_s$ |
| **Output:** grasp hypotheses $H$ of $v$ |
| $H = \emptyset$; |
| for $\theta \leftarrow 0$ to $\pi$ with interval $\theta'$ do |
| for $\phi \leftarrow 0$ to $2\pi$ with interval $\phi'$ do |
| $r^* \leftarrow \arg\max_{r} \{ (r, \theta, \phi) \in M \};$ |
| for $r \leftarrow 0$ to $r^*$ with interval $r'$ do |
| $h \leftarrow \text{GenerateHypothesis}(P_s, P_r, r, \theta, \phi);$ |
| if $\text{CollisionCheck}(h, M) = false$ then |
| Insert $h$ into $H$; |

Zhao et al.3
Grasp stability analysis

Inspired by force balance, we propose a metric to evaluate grasp stability for a two-finger parallel gripper. As shown in Figure 3, we establish a set of line segments connecting corresponding points between the two fingers. They are parallel with \( \mathbf{P}_m \), which is the movement direction of the fingers. \( \mathbf{P}_1 \) and \( \mathbf{P}_2 \) are the intersection points of the line segments and the object surface, which are closest to finger 1 and finger 2. \( \mathbf{N}_1 \) and \( \mathbf{N}_2 \) are surface normals at \( \mathbf{P}_1 \) and \( \mathbf{P}_2 \).

First, all grasp configurations of the objects that have been recognized in the scene are transferred into robot base frame according to the objects' poses. Then for each object, its grasp configurations in the upper one-fourth spherical space toward the robot are selected as its grasp candidates, since the supporting table and manipulator limitation prevent grasping from bottom and back. As shown in Figure 4, any selected grasp candidate \( \mathbf{P} \) of object \( O \) must satisfy

\[
\theta(\mathbf{d}, \mathbf{RO}_H) \leq \pi/2 \quad \text{and} \quad \angle(\mathbf{d}, \mathbf{V}_D) \leq \pi/2
\]

where \( \mathbf{RO}_H \) is the projection in the horizontal plane of the vector from the origin of robot base frame to the origin of object frame and \( \mathbf{V}_D \) is a vector pointing straight down.

For each grasp candidate \( \mathbf{P} \), its relative pose score is defined as follows

\[
E(\mathbf{P}) = K_s \cos \theta(\mathbf{d}, \mathbf{RO}_H) - K_d |\mathbf{RP}| \quad (4)
\]
where \( \mathbf{RP}_c \) is the vector from the origin of robot base frame to the grasp center point \( \mathbf{P}_c \), and \( K_0 \) and \( K_d \) are positive weight coefficients. The first term in the right of equation (4) represents the angle that the gripper has to turn to reach \( \mathbf{P} \), and the second term represents the distance that the gripper has to move to reach \( \mathbf{P} \). As the angle or distance is smaller, \( E(P) \) is larger, which indicates the gripper can reach \( \mathbf{P} \) more easily without consideration of obstacles.

**Obstruction analysis**

We utilize the distance map\(^{10} \) to identify obstruction relationships between grasp candidates and objects. We build a set of rays \( \{ \mathbf{r} \} \) constrained by \( \angle(-\mathbf{P}_d, \mathbf{r}) \leq \varepsilon \), where \( \varepsilon \) denotes the range to estimate free space around a grasp candidate. The first ray \( \mathbf{r}_0 \) is built according to Figure 5. Then rays \( \{ \mathbf{r}_i | i = 1, 2, \ldots \} \) are built by rotating \( \mathbf{r}_0 \) around \( \mathbf{P}_m \) with angle \( i\varepsilon' \leq \varepsilon \), where \( \varepsilon' \) is a constant interval. Finally, rays \( \{ \mathbf{r}_j | j = 1, 2, \ldots \} \) are built by rotating each \( \mathbf{r}_i \) around \( -\mathbf{P}_d \) with angle \( j\varepsilon < 2\pi \). All above rays constitute the set \( \{ \mathbf{r} \} \). The distance from grasp candidate \( \mathbf{P} \) to object \( O \) within angle \( \varepsilon \) is defined as follows

\[
D_i(P, O) = \min_{|\mathbf{r}|} \text{Dist}(\mathbf{r}, O)
\]  

where \( \text{Dist}(\mathbf{r}, O) \) indicates the distance from \( \mathbf{P}_0 \) to the nearest point on \( O \) that \( \mathbf{r} \) hits. According to our experience, \( \varepsilon \) had better be less than \( \pi/6 \), or the distance may be too conservative.

Then the OD of \( P \) by \( O \) is defined as follows

\[
\Delta(P, O) = \alpha^{D_i(P, O)}
\]  

where \( 0 < \alpha < 1 \) is a constant. If none of the rays hits \( O \), that is, \( D_i(P, O) = \infty \), then \( \Delta(P, O) = 0 \), which means \( O \) does not obstruct \( P \) at all. If \( O \) is in contact with \( \mathbf{P}_0 \), that is, \( D_i(P, O) = 0 \), then \( \Delta(P, O) = 1 \), which means \( O \) absolutely obstructs \( P \). Obviously, an object never obstructs its own grasp candidates. Generally, \( \alpha \) can be set such that \( \alpha^{D_{\text{max}}} < 0.1 \), where \( D_{\text{max}} \) is some distance between two objects considered not to obstruct each other.

Suppose \( \mathcal{O} = \{ O_i | i \in [1, n] \} \) are the \( n \) objects that have been recognized in the scene. \( B'(\mathcal{O}) \) is the set of grasp candidates of object \( O_i \). We define the OD of \( O_i \) by \( O_j \) as follows

\[
\Omega(O_i, O_j) = \sum_{P \in B'(\mathcal{O})} \Delta(P, O_j) / |B'(\mathcal{O})| \]  

where \( \Omega(O_i, O_j) \) is also between 0 and 1, the larger it is, and \( O_i \) causes more obstruction of grasping \( O_j \). \( \Omega = \{ \Omega(O_i, O_j) | i, j \in [1, n] \} \) is defined as object obstruction set of the current scene, which includes all ODs between each two objects.

**Grasp order planning**

Suppose \( \mathbf{X}_n = [x_1, \ldots, x_i, \ldots, x_n] \) represents grasp order of all objects in \( \mathcal{O} \). \( x_i \) is the object in the \( i \)th grasp, and \( x_i \neq x_j \) if \( i \neq j \). For a given \( \mathbf{X}_n \), we can array the elements in object obstruction set \( \Omega \) as an obstruction matrix

\[
S(\mathbf{X}_n) = \begin{bmatrix}
0 & \Omega(x_1, x_2) & \cdots & \Omega(x_1, x_n) \\
\Omega(x_2, x_1) & 0 & \cdots & \Omega(x_2, x_n) \\
\vdots & \vdots & \ddots & \vdots \\
\Omega(x_n, x_1) & \Omega(x_n, x_2) & \cdots & 0
\end{bmatrix}
\]  

\[
z(x_i) = \sum_{j=1}^{n} \Omega(x_i, x_j) \]

is called scene OD of \( x_i \), which indicates the whole OD by the other objects when grasping \( x_i \). The objective of grasp order planning is minimizing the sum of the scene ODs of all objects under grasp order \( \mathbf{X}_n \), that is

\[
\min Z(\mathbf{X}_n) = \min \sum_{i=1}^{n} z(x_i) = \min \sum_{i=1}^{n} \sum_{j=1}^{n} \Omega(x_i, x_j)
\]  

It actually minimizes the sum of the upper right part of the matrix \( S(\mathbf{X}_n) \), and the bottom-left part is ignored because the objects that have been removed no longer obstruct remaining objects.

This problem is similar to the traveling salesman problem (TSP), and any planning method, which can solve the TSP,
can also solve our problem. In our implementation, branch and bound method is utilized as shown in Algorithm 2.

The branch and bound method incrementally builds a tree to search for an optimal plan. A node \( q \) on the tree contains a grasp order \( q.order = \{x_1, \ldots, x_i, \ldots, x_m\} \) for grasping \( m \) objects, \( m \leq n \). Each node also has a lower bound \( q.lowerbound \), and its value is computed by

\[
Z_0(X) = \sum_{m=1}^{m} \sum_{j=i}^{n} O(x_i, x_j)
\]

\( X_{n-1} \) to \( x_n \) are objects in \( \Omega - q.order \), and the order is arbitrary.

At the beginning, an upper bound is initialized by a greedy method, which chooses the next object to be grasped with the least scene OD. Then a set \( Q \) is created to save all expandable leaf nodes on the tree, it initially contains a single root node \( q_0 \), which includes no object. At each loop, the leaf node \( q \in Q \), which has the lowest lower bound, is taken out to be expanded. Each object, which is not in \( q.order \), is added to create a new node \( q' \), and the lower bound is updated. If the new node’s lower bound is less than the upper bound, it is inserted in \( Q \) as a child of \( q \). Moreover, if \( q'.lowerbound \) already includes all the objects, the upper bound is updated to \( q'.lowerbound \), and in this condition, \( q' \) will not be inserted in \( Q \) as it cannot be further expanded. The loop repeats until there is no node can be expanded or the lowest lower bound is larger than the upper bound.

At the beginning, an upper bound is initialized by a greedy method, which chooses the next object to be grasped with the least scene OD. Then a set \( Q \) is created to save all expandable leaf nodes on the tree, it initially contains a single root node \( q_0 \), which includes no object. At each loop, the leaf node \( q \in Q \), which has the lowest lower bound, is taken out to be expanded. Each object, which is not in \( q.order \), is added to create a new node \( q' \), and the lower bound is updated. If the new node’s lower bound is less than the upper bound, it is inserted in \( Q \) as a child of \( q \). Moreover, if \( q'.lowerbound \) already includes all the objects, the upper bound is updated to \( q'.lowerbound \), and in this condition, \( q' \) will not be inserted in \( Q \) as it cannot be further expanded. The loop repeats until there is no node can be expanded or the lowest lower bound is larger than the upper bound.

Algorithm 2. GraspOrderPlanning.

Input: object set \( X \), object obstruction set \( \Omega \)
Output: grasp order \( X_n \)
1. \( X_n \leftarrow \text{GreedMethod} \)
2. \( upperbound \leftarrow Z(X_n) \)
3. \( Q \leftarrow \emptyset \)
4. \( q_0.order \leftarrow \emptyset \)
5. \( Q.insert(q_0) \)
6. while \( Q \neq \emptyset \) do
7. \( \text{Select } q \in Q \text{ which has the lowest lowerbound; } \)
8. if \( q.lowerbound > upperbound \) then
9. \( \text{ break; } \)
10. \( \text{forall } O \in (\Omega - q.order) \text{ do } \)
11. \( q' \leftarrow q \)
12. \( q'.order.pushback(O) \)
13. \( q'.lowerbound \leftarrow Z'(q'.order) \)
14. if \( q'.lowerbound < upperbound \) then
15. \( \text{if } q'.order.size < n \text{ then } \)
16. \( Q.insert(q' \)
17. \( \text{ else } \)
18. \( upperbound \leftarrow q'.lowerbound \)
19. \( X_n \leftarrow q'.order \)
20. \( Q.delete(q) \)

Grasp configuration planning

After planning the grasp order, we should decide which grasp candidate is used to grasp each object. Suppose \( Y_n = \{y_1, \ldots, y_i, \ldots, y_n\} \) represents grasp configurations to grasp objects in \( X_n \), \( y_i \) is used to grasp \( x_i \). For a grasp candidate \( P \) of \( x_i \), its relative pose score under obstruction is defined as follows

\[
E'(P) = E(P) - \sum_{j=i}^{n} \Delta(P, x_j)
\]

where \( E'(P) \) estimates how easily the gripper can reach \( P \) with consideration of obstacles.

For object \( x_i \), its final grasp configuration is computed by Algorithm 3. A motion planner is required to compute
collision-free trajectories of the robot. If there is at least one trajectory that arrives at some grasp configuration, then this grasp configuration is reachable. First, \( y_i \) is given a default \( P_0 \), which cannot be executed, and \( B'(x_i) \) is ranked in descending order with respect to \( E'(P) \). Then motion planning is called for each \( P \in B'(x_i) \), until the first reachable one is found.

The complete \( Y_n \) can be decided at one time according to \( X_n \) before any object is grasped. But the remaining objects’ poses may change after each grasp due to accidental collision with the robot or removing an object which supports some others. In addition, previously invisible objects may be revealed after each grasp. So we implement the whole planning process as Algorithm 4. At one time, only one object’s grasp configuration is planned, and the object is grasped immediately. Then grasp order planning and grasp configuration planning will be called again. The grasp order of the remaining objects may change because of the change of objects’ poses or the discovery of new objects. If no grasp candidate of an object is reachable, the next object in \( X_n \) will be tried. In the worst case, no object can be grasped (line 11), at this time the task is aborted.

**Complexity analysis**

Suppose one object has at most \( m \) grasp configurations, that is, \( B'(O) \leq B(O) \leq m \). In obstruction analysis, Dist\( (r, O) \) has to be computed at most \( mA^2 \) times, the time complexity is \( O(n^2 - n) \). For a whole task, replanning is done after grasping each object, so the total complexity is \( O(\sum_{k=2}^{n} (k^2 - k)) = O(n^3 - n) \), which runs in polynomial time.

Grasp order planning cannot promise to run in polynomial time, as in the worst-case \( n! \) grasp orders may be explored. Generally, the branch and bound method is fairly efficient for small-scale instances. For large amount of objects, we can divide objects into several groups according to their poses and plan grasp order for each group, respectively.

In grasp configuration planning, at most \( mn \) grasp configurations are tested by motion planner for \( n \) objects, thus the time complexity is \( O(n) \). Although it looks low, grasp configuration planning actually occupies the most part in the whole planning time, because motion planning itself is time-consuming.

**Simulation and experiment**

**Setup**

We built an experiment platform as shown in Figure 7(a). The robot is an ABB IRB 120 manipulator with a Robotiq 2F-140 gripper. A Microsoft Kinect is mounted on the manipulator. A small table with the size of 40 cm × 40 cm is in front of the robot. We also built a simulation platform.
platform as shown in Figure 7(b) based on Gazebo, and the virtual robot is the same as the real robot.

Six objects and their IDs are shown in Figure 8. We measured the sizes of these objects and directly modeled them as cylinders or boxes. Approximate models are used not only for grasp generation, obstruction analysis, and collision checking but also for grasping in simulations.

The parameters for grasp generation are $\theta = \pi/4$, $\phi' = \pi/6$, $r' = 4$ cm, $P_\mathcal{R} = \pi/4$, $\lambda_1 = \lambda_2 = \lambda_3 = 1$, and $M_\gamma = 0.15$. The parameters for grasp planning are $\varepsilon = \pi/9$, $\varepsilon' = \pi/18$, $\alpha = 0.01$, and $K_\phi = K_\delta = 1$. Rapidly-Exploring Random Tree (RRT)-connect integrated into MoveIt is adopted as the motion planner. If motion planning run time reaches 1 s and no collision-free trajectory is found, the grasp candidate is regarded as unreachable.

We would like to compare our planning method based on OD with a baseline method based on grasp ranking (GR), which ranks all grasp candidates by some metric and selects the best one. Here, we use the relative pose score as the metric and Algorithm 5 shows the details. Grasp candidate selection is the same as our method, then $B' \subseteq \bigcup_{x \in \mathcal{Q}, y \in B'(x)} \{x\}$ is collected and sorted in descending order with respect to relative pose score $E$. Motion planning is tested for each grasp candidate until the first reachable one is found and the robot immediately grasps an object. These procedures are repeated until no object can be grasped. The GR method is satisfactory at the most times for grasping the single object, and we hope to see the performance of the two methods for grasping multiple cluttered objects.

To evaluate performance, the following data are recorded during simulations and experiments:

1. **Number of picked objects**: the number of objects that are successfully grasped.
2. **Number of fallen objects**: the number of objects that fall from their original poses.
3. **Motion time**: the time that is spent on robot motion.
4. **Obstruction analysis time**: the time that is spent on obstruction analysis.

---

**Algorithm 5. Grasp Ranking based Planning.**

1. $\mathcal{Q} \leftarrow \text{Object Recognition}$;
2. while $\mathcal{Q} \neq \emptyset$ do
3.   $B' \leftarrow \text{Grasp Candidate Selection} (\mathcal{Q}, B)$;
4.   Rank $B'$ in descending order with respect to $E(P)$;
5.   $y \leftarrow P_0$;
6.   foreach $P \in B'$ do
7.     if $\text{Motion Planning} (P) = \text{success}$ then
8.       $y \leftarrow P$;
9.       break;
10.    if $y \neq P_0$ then
11.       Grasp $x$ by $y$; // $x \in \mathcal{Q}, y \in B'(x)$
12.    else
13.       return;
14.   $\mathcal{Q} \leftarrow \text{Object Recognition}$;

---

**Figure 8.** Objects and IDs. 3D models are shown under the real objects. (a) object 1, (b) object 2, (c) object 3, (d) object 4, (e) object 5, and (f) object 6. ID: identification; 3D: three-dimensional.

**Figure 9.** Examples of simulation scenes.
5. **Grasp order planning time**: the time that is spent on grasp order planning.

6. **Grasp configuration planning time**: the time that is spent on grasp configuration planning.

7. **Number of motion planning trials**: the number of grasp candidates that are tested by motion planning.

**Simulation**

We generated four sets of scenes for 3, 4, 5, and 6 objects, respectively, and each set has 50 random scenes. To generate a scene, a first certain number of objects are randomly placed. Table 1 shows the simulation results, and Table 2 shows the average data per successful grasp.

### Table 1. Simulation results.

| Scene set   | Method | Number of picked objects | Number of fallen objects | Motion time (s) | Obstruction analysis time (s) | Grasp order planning time (s) | Grasp configuration planning time (s) | Number of motion planning trials |
|-------------|--------|--------------------------|--------------------------|-----------------|------------------------------|-------------------------------|----------------------------------|----------------------------------|
| Three objects | OD     | 147                      | 0                        | 3205.15         | 0.2582                       | 2.24e–3                       | 78.48                           | 440                               |
|             | GR     | 140                      | 7                        | 3057.15         | —                            | —                            | 77.83                           | 570                               |
| Four objects | OD     | 197                      | 0                        | 4319.71         | 0.6296                       | 5.14e–3                       | 94.74                           | 634                               |
|             | GR     | 177                      | 20                       | 3892.72         | —                            | —                            | 80.76                           | 896                               |
| Five objects | OD     | 244                      | 0                        | 5355.70         | 1.1340                       | 9.68e–3                       | 125.61                          | 751                               |
|             | GR     | 220                      | 29                       | 4858.00         | —                            | —                            | 144.18                          | 1221                              |
| Six objects | OD     | 293                      | 2                        | 6395.54         | 2.0693                       | 20.9e–3                       | 146.42                          | 907                               |
|             | GR     | 261                      | 38                       | 5738.23         | —                            | —                            | 148.72                          | 1502                              |

Source: Each datum is the sum of completing all tasks in the 50 scenes. OD: obstruction degree; GR: grasp ranking.

### Table 2. Average data per successful grasp.

| Scene set   | Method | Planning time (s) | Motion time (s) | Number of motion planning trials |
|-------------|--------|-------------------|-----------------|---------------------------------|
| Three objects | OD     | 0.5357            | 21.80           | 2.99                            |
|             | GR     | 0.5559            | 21.84           | 4.07                            |
| Four objects | OD     | 0.4841            | 21.93           | 3.22                            |
|             | GR     | 0.4563            | 21.99           | 5.06                            |
| 5 objects   | OD     | 0.5195            | 21.95           | 3.08                            |
|             | GR     | 0.6553            | 22.08           | 5.55                            |
| 6 objects   | OD     | 0.5069            | 21.83           | 3.10                            |
|             | GR     | 0.5698            | 21.99           | 5.75                            |

OD: obstruction degree; GR: grasp ranking.

**Figure 10.** (a) and (b) Scenes causing object falling under method OD. OD: obstruction degree.

**Figure 11.** Scenes for experiments. Left part of each subfigure shows the arrangement of objects. Right part of each subfigure shows the result of object recognition before the first grasp. (a) Scene 1, (b) scene 2, (c) scene 3, and (d) scene 4.
selected, then they are dropped one by one from random positions over the table. Figure 9 shows some examples of the scenes. In the simulation, the robot can completely get IDs and poses of all objects on the table. Each scene was tested once under methods OD and GR, respectively, and the results are presented in Table 1. The source code for simulation is available at https://github.com/Kazfyx/grasp-planning.

Unsuccessful grasping may be due to two reasons: (1) the object is initially at an unreachable pose, that is, all of its grasp candidates are not reachable, and (2) the object falls and rolls to an unreachable pose. We can see that more objects were grasped under method OD since less objects fell down. Method GR caused more fallen objects because it is originally designed for grasping a single object and no appropriate order of grasping is planned. As the number of objects increased, more objects fell because complex object interactions occurred more often.

Planning time (including time of obstruction analysis, grasp order planning, and grasp configuration planning) increased as the number of objects increased. Obstruction analysis time conformed to the time complexity $O(n^3 \cdot n)$. Grasp order planning also approximately ran in polynomial time, which proves the efficiency of the branch and bound method. Grasp configuration planning took the most time as a lot of unreachable grasp candidates were tested.

Table 2 presents some average data computed from Table 1. The average planning time per grasp has no obvious difference between the two methods, in the condition that GR does not do obstruction analysis and grasp order planning. Less motion planning trials were taken in OD since obstruction analysis gives which object is more likely to be graspable and which grasp candidate is more likely to be reachable. Besides, the average motion time per grasp is nearly unchanged over all simulations. Because the grasp candidates that likely generate short paths are tested first, and object positions did not vary much on a not big table.

Infrequently, there were two fallen objects under method OD. In Figure 10(a), object 1 sat on object 6 while leaned against object 4. Slanting angle of object 1 was too little to filter grasp configurations toward object 4, so object 4 was grasped first as it had more unobstructed grasp candidates. In Figure 10(b), object 2 leaned against object 6 from the back, and object 6 was not obstructed at all and was grasped first. In this case, object 2 can never be grasped first as it was unreachable.

**Experiment**

We designed four scenes as shown in Figure 11 for experiments. Object recognition is based on 3D point cloud captured by the Kinect, and occluded objects may not be recognized. Scene 1 represents the situation that objects lean against others, scene 2 represents the situation that objects sit on others, and scenes 3 and 4 are complex situations including multiple object interactions.
Each scene was tested once under methods OD and GR, respectively, and the results are presented in Table 3. Method OD picked all the objects in the four scenes with no object collapse and taken less time for planning. Method GR led to object collapse in 3 scenes, 3 objects fell down and were not able to be grasped later.

In scene 1, all objects were recognized at the beginning. Figure 12 shows the selected grasp candidates, and Table 4 presents the object obstruction set. The execution processes are shown in Figure 13. OD planned a grasp order of $X_4 = \{\text{object 2}, \text{object 1}, \text{object 3}, \text{object 4}\}$. GR grasped object 4 before object 3 because object 4 had a reachable grasp candidate with the highest relative pose score among all grasp candidates of objects 3 and 4. Then object 3 fell and came to an unreachable pose, so it could not be grasped later. In this case, much time was spent on grasp configuration planning as all grasp candidates of object 3 were tested by motion planning.

Figure 13. Execution processes in scene 1 under methods (a) OD and (b) GR. OD: obstruction degree; GR: grasp ranking.
Conclusion and discussion

This article proposes a grasp planning method based on OD. First, a grasp configuration database, which has widely distributed grasp approach directions, is built and then it facilitates the search of reachable grasp configurations in cluttered scenes. Then ODs of grasp configurations and objects are analyzed according to the geometric relation between grasp configurations and object models. Finally, grasp order is planned in which object interactions are implicitly inferred by the ODs. At the same time, reachable grasp configuration is searched quickly for each object. Simulations and experiments in a series of scenes demonstrate that our method can grasp objects in clutter efficiently and safely.

Compared with previous grasp action planning, we consider the relationships among objects and give proper order of grasping, which reduces the risk of object collapse. Compared with previous work on object support relation recognition, our method does not need complex computer vision algorithms or mechanical analysis and plans concrete grasp configurations. Also because of that, it cannot completely avoid improper grasp order, as object interaction is not explicitly recognized.

In the future, we are going to add more constraints in grasp planning and introduce risk estimation to allow the robot to terminate task execution when object collapse may happen. In addition, the uncertainty of perception is common, so possible unseen objects are also going to be taken into account for grasp order planning.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work is supported by the National Key R&D Program of China under grant 2017YFB1303600.

ORCID iDs

Wenrui Zhao https://orcid.org/0000-0003-4976-9518
Weidong Chen https://orcid.org/0000-0001-8757-0679

Supplemental material

Supplemental material for this article is available online.

References

1. Yamanobe N and Nagata K. Grasp planning for everyday objects based on primitive shape representation for parallel jaw grippers. In: 2010 IEEE international conference on robotics and biomimetics, Tianjin, China, 14–18 December 2010, IEEE, pp. 1565–1570.
2. Huebner K and Kragic D. Selection of robot pre-grasps using box-based shape approximation. In: 2008 IEEE/RSJ international conference on intelligent robots and systems, Nice, France, 22–26 September 2008, IEEE, pp. 1765–1770.
3. Goldfelder C, Allen PK, Lackner C, et al. Grasp planning via decomposition trees. In: 2007 IEEE international conference on robotics and automation, Rome, Italy, 10–14 April 2007, IEEE, pp. 4679–4684.
4. Marton ZC, Goron L, Rusu RB, et al. Reconstruction and verification of 3D object models for grasping. In: Robotics research, Berlin, Heidelberg, 2011, pp. 315–328. Berlin, Heidelberg: Springer.
5. Harada K, Tsuji T, Nagata K, et al. Grasp planning for parallel grippers with flexibility on its grasping surface. In: 2011 IEEE international conference on robotics and biomimetics, Karon Beach, Thailand, 7–11 December 2011, IEEE, pp. 1540–1546.
6. Vahrenkamp N, Koch E, Wächter M, et al. Planning high-quality grasps using mean curvature object skeletons. IEEE Robot Autom Lett 2018; 3(2): 911–918.
7. Tsuji T, Uto S, Harada K, et al. Grasp planning for constricted parts of objects approximated with quadric surfaces. In: 2014 IEEE/RSJ international conference on intelligent robots and systems, Chicago, IL, USA, 14–18 September 2014, IEEE, pp. 2447–2453.
8. Li Y, Saut JP, Petitre J, et al. Fast grasping planning using cord geometry. IEEE Trans Robot 2015; 31(6): 1393–1403.
9. Wan W, Harada K, and Kamehiro F. Planning grasps with suction cups and parallel grippers using superimposed segmentation of object meshes. IEEE Trans Robot 2021; 37(1): 166–184.
10. Berenson D, Diankov R, Nishiwaki K, et al. Grasp planning in complex scenes. In: 2007 IEEE-RAS international conference on humanoid robots, Pittsburgh, PA, USA, 29 November–1 December 2007, IEEE, pp. 42–48.
11. Dogar MR, Hsiao K, Ciocarlie M, et al. Physics-based grasp planning through clutter. In: Robotics: science and systems VIII, Cambridge, MA, 2013, pp. 57–64. Cambridge, MA: MIT Press.
12. Muhayyuddin, Moll M, Kavraki L, et al. Randomized physics-based motion planning for grasping in cluttered and uncertain environments. IEEE Robot Autom Lett 2018; 3(2): 712–719.
13. Wei Z, Chen W, Wang H, et al. Manipulator motion planning using flexible obstacle avoidance based on model learning. Int J Adv Robot Syst 2017; 14(3): 1729881417703930.
14. Eppner C and Brock O. Planning grasp strategies that exploit environmental constraints. In: 2015 IEEE international conference on robotics and automation, Seattle, WA, USA, 26–30 May 2015, IEEE, pp. 4947–4952.
15. Elliott S, Valente M, and Cakmak M. Making objects graspable in confined environments through push and pull manipulation with a tool. In: 2016 IEEE international conference on robotics and automation, Stockholm, Sweden, 16–21 May 2016, IEEE, pp. 4851–4858.
16. Stilman M, Schamburek JU, Kuffner J, et al. Manipulation planning among movable obstacles. In: 2007 IEEE
17. Dogar MR and Srinivasa SS. A planning framework for non-prehensile manipulation under clutter and uncertainty. *Auton Robot* 2012; 33(3): 217–236.

18. Panda S, Hafez AA, and Jawahar C. Learning support order for manipulation in clutter. In: *2013 IEEE/RSJ international conference on intelligent robots and systems*, Tokyo, Japan, 3–7 November 2013, IEEE, pp. 809–815.

19. Wu P, Chen W, Liu H, et al. Predicting grasping order in clutter environment by using both color image and points cloud. In: *2019 WRC symposium on advanced robotics and automation*, Beijing, China, 21–22 August 2019, IEEE, pp. 197–202.

20. Mojtahedzadeh R, Bouguerra A, Schaffernicht E, et al. Support relation analysis and decision making for safe robotic manipulation tasks. *Robot Auton Syst* 2015; 71: 99–117.

21. Kartmann R, Paus F, Grotz M, et al. Extraction of physically plausible support relations to predict and validate manipulation action effects. *IEEE Robot Autom Lett* 2018; 3(4): 3991–3998.

22. Fu Y, Li M, Zhang X, et al. Spatial topological relation analysis for cluttered scenes. *Sensors* 2020; 20(24): 7181.

23. Lippiello V, Ruggiero F, Siciliano B, et al. Visual grasp planning for unknown objects using a multifingered robotic hand. *IEEE/ASME Trans Mechatron* 2012; 18(3): 1050–1059.

24. Lei Q and Wisse M. Fast grasping of unknown objects using force balance optimization. In: *2014 IEEE/RSJ international conference on intelligent robots and systems*, Chicago, IL, USA, 14–18 September 2014, IEEE, pp. 2454–2460.

25. Lin HC, Tang T, Fan Y, et al. A framework for robot grasp transferring with non-rigid transformation. In: *2018 IEEE/RSJ international conference on intelligent robots and systems*, Madrid, Spain, 1–5 October 2018, IEEE, pp. 2941–2948.

26. Lenz I, Lee H, and Saxena A. Deep learning for detecting robotic grasps. *Int J Robot Res* 2015; 34(4–5): 705–724.

27. Levine S, Pastor P, Krizhevsky A, et al. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *Int J Robot Res* 2018; 37(4–5): 421–436.

28. Fang HS, Wang C, Gou M, et al. GraspNet-1Billion: a large-scale benchmark for general object grasping. In: *2020 IEEE/CVF conference on computer vision and pattern recognition*, Seattle, WA, USA, 13–19 June 2020, IEEE, pp. 11444–11453.

29. León B, Morales A, and Sancho-Bru J. Human grasp evaluation. In: *From robot to human grasping simulation*, Cham, 2014, pp. 175–206. Cham: Springer.

30. Tagliasacchi A, Alhashim I, Olson M, et al. Mean curvature skeletons. In: *2012 Computer graphics forum*, volume 31. Hoboken, NJ, pp. 1735–1744. Hoboken, NJ: Wiley Online Library.

31. Clausen J. Branch and bound algorithms—principles and examples. *Technical Report, Department of Computer Science, University of Copenhagen, Denmark*, 1999, pp. 1–30.

32. Kuffner JJ and LaValle SM. RRT-connect: an efficient approach to single-query path planning. In: *2000 IEEE international conference on robotics and automation*, volume 2, IEEE, San Francisco, CA, USA, 24–28 April 2000, pp. 995–1001.

33. Bohg J, Morales A, Asfour T, et al. Data-driven grasp synthesis—a survey. *IEEE Trans Robot* 2014; 30(2): 289–309.

34. Zheng L, Wang H, and Chen W. A fast 3D object recognition pipeline in cluttered and occluded scenes. In: *2017 international conference on intelligent robotics and applications*, Wuhan, China, 16 August 2017, Springer, pp. 588–598.