Received: 6 February 2018 | Revised: 9 August 2018 | Accepted: 9 August 2018
DOI: 10.1002/cav.1861

RESEARCH ARTICLE

WILEY

Disassembling a 3D mechanism for efficient packing

Mingyuan Li[1] | Xiaoheng Jiang[1] | Ningbo Gu[1] | Weiwei Xu[2] | Junxiao Xue[1] | Bing Zhou[1] | Mingliang Xu[1]

[1]School of Information Engineering, Zhengzhou University, Zhengzhou, China
[2]College of Computer Science and Technology, Zhejiang University, Hangzhou, China

Correspondence
Mingliang Xu, School of Information Engineering, Zhengzhou University, No. 100 Science Avenue, 450001 Zhengzhou, China.
Email: iexumingliang@zzu.edu.cn

Funding information
National Science Foundation of China, Grant/Award Number: 61672469, 61472370, 61822701, and 61732016; Science & Technology Innovation Talents in Universities of Henan Province, Grant/Award Number: 18HASTIT020; Young Scholar in Universities of Henan Province, Grant/Award Number: 2016GGJS-001; National Key Technology Research and Development Program of China, Grant/Award Number: 2017YFC0804401; Alibaba IDEA Lab and fundamental research fund for the central universities, Grant/Award Number: 2017YFB1002600

Abstract

This paper introduces a disassemble-and-pack algorithm to disassemble a mechanical 3D model in groups that can be efficiently packed within a box, with the objective of reassembling them easily after delivery. Its key feature is that, mostly, the mechanism can be disassembled at the joint and each part can be an adjusted motion structure based on its joint type. Our system consists of two steps: disassembling the mechanical object into a group set and packing them within a box efficiently. The first step consists in the creation of a hierarchy of possible group set of parts that can be tightly packed within their minimum bounding boxes. Use the breadth-first search algorithm to traverse the hierarchy of possible group set in order to disconnect the joints and get the group set. In the second step, according to the reverse order of volume, each group in the set is inserted into the specified box. The fact that mechanism disassembly and shape packing are both an NP-complete problem justifies finding approximated solutions according to efficacy and efficiency. Experimental results show that our approach can really efficiently pack a range of mechanisms from a simple model to complex objects.

KEYWORDS
disassembly, mechanism, pack

1 | INTRODUCTION

Three-dimensional mechanism fabrication is one of the central tasks in commercial CAD software, and its impact on both academia and industry is so profound. Disassembly and packing are two of the well-studied optimization problems in mechanism fabrication. In a typical technique, disassembly seeks to disconnect a mechanical object into a group set of parts, with each satisfying a desirable geometric property, and packing involves placing a group set into a given container. With the fact that two problems are an NP-complete problem, heuristics and approximated solutions must be used, and algorithm efficiency is an important result.

In this paper, we present an approach to solve the two previously studied problems: mechanism disassembly and bin packing problem.[1,2] In the first step, the mechanical object is disassembled into a group set of parts with total minimum volume. It means that the joint with *maximum cost* must be disconnected and adjusted to every part in each group. In the second step, our approach computes a roto-translation for each of the groups when inserted into a specified box. The major contribution of this paper is in how to disassemble a mechanical object into a group set that can be packed efficiently.

Our original contributions can be summarized as follows:

- a novel disassemble-and-pack integrated approach to disconnect a minimum number of joints and pack efficiency;
- a new hierarchical disassembly algorithm that produces a group set with minimum total volume;
- a minimum oriented bounding box (OBB) of the mechanical group algorithm that calculates the motion parameters of the parts.

A preliminary version of this paper appeared in the work of Li et al.[3] This paper extends the earlier work[3] as follows. First, in order to improve the calculation speed, compared with the previous work using a triangular mesh representation model, we voxelize each part of the mechanical object to represent an original mesh in 3D. Voxelization is applied in collision detection, motion parameter optimization, and packing algorithm. Second, with the fact that shape packing is an NP-complete problem, the group insertion order is from a large volume of the part to a small one. Experiments show that the insertion sequence can improve the space utilization of the packing strategy. Third, comparing our method with state-of-the-art methods, such as the splitPack,[2] in space utilization and time efficiency, our algorithm has a significant improvement. Fourth, the key parameters of the bottom surface of the box are analyzed, and the appropriate setting strategy is obtained.

This paper is organized as follows. We summarize related work in Section 2. Section 3 describes the notation of our algorithm and the overall disassemble-and-pack approach. Section 4 describes how to disassemble a mechanism to a group set in a hierarchical strategy based on joint cost. In Section 5, we propose a novel mechanical group packing algorithm. Section 6 discusses the test results for the four examples of the mechanism. Finally, we conclude this paper with a summary and discussion of future work in Section 7.

2 | RELATED WORK

Mechanism disassembly. Our algorithm is applied to the mechanical model. The goal of mechanism disassembly based on joint type is to part a mechanical object to a group set that has minimum total volume of all group OBBs. A comprehensive disassembly approach[4] automatically generates a disassembly sequence from a hierarchical attributed liaison graph. However, in order to promote space utilization, the structure motion of mechanical parts should be take into account. For a complex mechanism with multiple parts, some approaches[5–8] visualize its possible motion. As pointed out in the work of Xu et al.,[9] the relative motion of two parts connected via a joint is based on joint type. According to some present approach, the joint type information can be automatically obtained or manually specified. For instance, the modeling approach in the work of Zhu et al.[10] automatically generates a mechanism assembly located in a box below the feature base that produces the specified motion, and another modeling approach in the work of Xu et al.[11] focuses on the shapes of mechanical part primitives and their motion constraints in order to obtain both geometry and structure via multiview images. Different from the study of Xu et al.,[12] our approach uses a greedy strategy to exploit the hierarchy of possible group set for high efficiency.

Mechanism packing. The goal of the algorithm is to solve a 3D bin packing problem for a mechanical object. We hope to minimize the volume of a specified box containing a set of appropriately placed objects. In order to avoid the NP-complete problem,[13] heuristic-driven algorithms exist to find approximated solutions. The Cutting and Packing approach,[14] the Decomposing and Packing approach,[1] and the Splitting and Packing approach[2] can be summarized in two main steps: partitioning a given object and placing a set of parted objects into a specified container. In the study of Bansal et al.,[15] the base area of the container is fixed, and the focus of the approach is to minimum height. Our work is mostly related to the Splitting and Packing approach.[2] However, different from splitting object in the work of Attene,[2] our work focuses on the disassembling of the mechanical object into parts and adjusting each part to the minimum volume of an OBB for efficient packing. The previous work seeks to decompose the static model into a group of static parts that can be efficiently packed. However, for the mechanical model, an arbitrary decomposition model can cause the model to fail to assemble. The mechanical model should be disassembled at its joint. Instead of the static part, each mechanical part's degree of freedom has been changed after disassembly.

3 | DISASSEMBLE-AND-PACK APPROACH

In this section, we describe the overview of our algorithm: mechanism disassembly and mechanical group packing. We begin by fixing our notation involved in the algorithm.

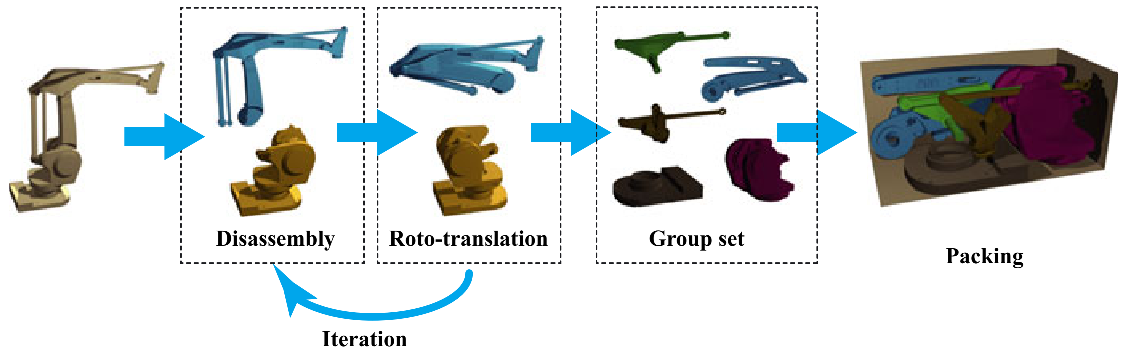

FIGURE 1 Overview. The initial mechanical robot-arm is disassembled into two groups. Rotate and translate each part of the groups to minimize the volume of every group. Repeat the above two steps to obtain the group set with the satisfied disassembling efficiency. Insert the groups into the specified box one by one

3.1 | Notation

Mechanical notation. Our algorithm treats a mechanism object as a collection of rigid parts interconnected to transmit rigid motions, and a joint connects two parts to form a kinematic pair. Let P denote the part set of the mechanical object and $P = \{p_0, p_1, \dots, p_{N_p}\}$, where N_p is the number of the mechanical object's part. A joint connects two parts p_i and p_j indicated by $c_{(p_i,p_j)}$. A joint c is binary: connect or disconnect. A group has one or more parts, and their joints are indicated by g. Not one joint between the arbitrary pair parts from two different groups is connected. Initially, a group set has only one group, which includes all mechanical parts. Disassembling the mechanical object, some joints being set disconnected and one group being split into several groups. A group set is denoted as $G = \{g_0, g_1, \dots, g_{N_g}\}$, where N_g is the number of groups.

Bounding box. A bounding box is usually used heuristically to speed up computation. In the bin packing problem, the calculation of bounding boxes is a fundamental problem. According to the bounding tightness required, the bounding box can be formulated in three ways: axis-aligned bounding boxes, OBBs, and minimum-volume bounding boxes (MBBs). Axis-aligned bounding boxes are aligned with the axes of the coordinate system; it can be computed very efficiently but not sufficiently. Principal component analysis can compute for OBBs. So far, the most outstanding is O'Rourke's,[16] but it is of high computational complexity ($O(n_3)$ for n input points). In order to calculate the exact volume, we use voxelization to calculate the volume. The bounding box volume consists of two kinds of voxels: one are voxels occupied by the object and the other are free voxels. The set of all voxels belonging to a part is called the voxel-box and denoted by Vox(p).

3.2 | Overall approach

Our approach consists of two main steps, namely, disassembly and packing depicted in Figure 1. The goal of our algorithm is to pack the mechanical object into the specified box with high space utilization. The input to our process includes the mechanical object with its joint set and a target packing specified box. First, our algorithm disassembles the mechanical object to a group set based on the joint set. Then, it rotates and translates parts of every group so that they fit an axis-aligned box of minimum volume. In the group number being a constant number of the premise, the algorithm of disassembly expects to get the group set with the minimum sum of group OBB volume. Finally, the algorithm efficiently packs each group with rotated and translated parts into the box in sequence.

It is popular that the hierarchical approach disassembles the mechanical object (see, e.g., the work of Dong et al.[4]). In our paper, we employ a novel binary hierarchical of disassembling object approach to provide much tighter packing. In each disassembly step, a selected joint is set disconnected, and the mechanical object is disassembled into more groups. Then, every part in each group has more degrees of freedom to be translated and rotated. Since the object volume is constant, minimizing the group box volume means minimizing the hole volume of the group box. On the basis of joint type, translate and rotate the parts in each group to minimize the hole volume. After each disassembly step, we obtain a group set with minimum total volume of the group set, which will be packed into the box.

To minimize the number of disconnected joints, our approach employs breadth-first search for traversing a decision binary hierarchical tree. It is for minimizing the cost of disassembly and reassembly take. We started with analyzing the root of the tree (i.e., the whole object): If it can be packed into a specified box with sufficient efficiency (according to the

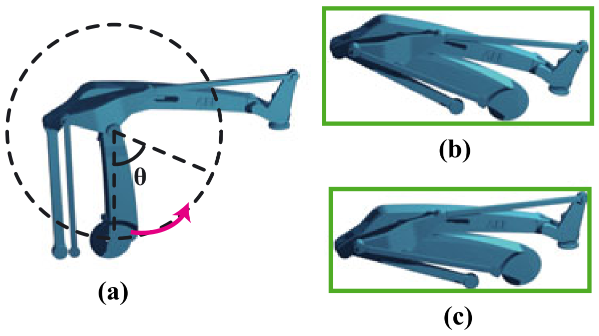

FIGURE 2 Motion parameter optimization. (a) A group of mechanical parts with motion parameters is a part of disassembled objects (disassembly group). (b) Optimize the motion parameters to minimize the volume of the group (roto-translation group). (c) Compute the minimum-volume bounding box (MBB) of the group (MBB group)

target), then we have a solution and stop. Otherwise, select a joint that is disassembled and can minimize the wasted space and disconnect it. The root is separated into two children, which represent the group of mechanical parts. Continue with packing and calculating the corresponding efficiency. If the result is satisfactory, the process stops; otherwise, our approach goes down further in the hierarchy and separates one node into two children, packs with best efficiency, and so on. If all joints have been disconnected, the process stops (see Figure 2).

The unique parameter of the algorithm is packing efficiency. It affects the number of joints that need disconnecting. The initial setting of the packing efficiency is also the termination condition of the traversing decision binary hierarchical tree. Unfortunately, object packing is an NP-hard problem, and not one approach can obtain the best packing efficiency and minimum number of disconnect joints. Traversing the hierarchical tree produces a lot of solution space; hence, we need to prune the tree to simplify the solution space. In fact, the greedy algorithm is adopted, which means that in each layer of the hierarchical tree, only one joint is selected, which can promote maximum efficiency and disconnect it. Experiments show that the simplified solution space can obviously improve the algorithm speed, and packing efficiency can be satisfied.

4 | HIERARCHICAL DISASSEMBLING BASED ON JOINT COST

At the beginning of our algorithm, our approach disassembles the mechanical object into several groups of parts and minimizes the sum of group MBB volume. There are two main problems that must be solved: one is how to calculate the volume of the group box and the other is which joint should be selected. To calculate the group box volume, the mechanical object must be actually a solid. This means that surface meshes with holes, self-intersections, and other sorts of defects must be processed in advance using appropriate mesh repairing tools.[17] *Voxelization* is used to calculate the volume of the group box, showed in Figure 3, and it also avoids the problem that the input mesh is not solid. To minimize the sum of group box volume, in each disassembling step, the joint with maximum *wasted volume cost* is selected. Disconnect the selected joint to generate more groups for packing.

4.1 | Voxels of part's volume

Calculating the volume of an object can be a time-consuming task. Accurately calculating the volume of the object is to sum up each volume of tetrahedron. However, this method requires transforming our mesh into a tetrahedral mesh. Unfortunately, our algorithm requires a lot of calculation collision volume of two parts. In order to calculate efficiency, we prefer to use simple data structures to calculate collision volume instead of the tetrahedral mesh. Therefore, our algorithm adopts voxelization to calculate the volume of the parts. Voxelization not only applies to almost all meshes but also can greatly improve the efficiency of the algorithm.

At beginning of our algorithm, we voxelize each part of the mechanical object to represent the original mesh in 3D. This is to speed up the calculation of volume and collision. Firstly, using principal component analysis, our algorithm computes the oriented bounding box OBB(p) of each part. On the basis of the three oriented axes of the OBB(p), the bounding box can be voxelized into several voxels in 3D. Then, the voxels are classified as either free or occupied, as depicted in Figure 3. The occupied voxels represent the shape of the mechanical part, and the number of occupied voxels

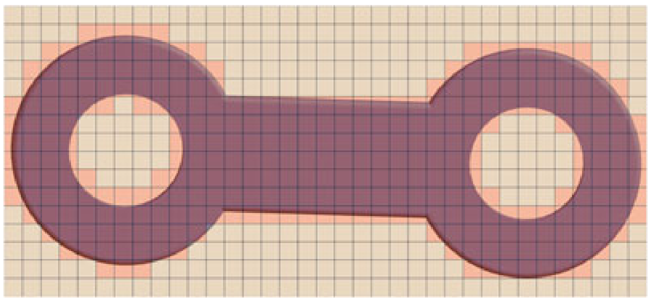

FIGURE 3 The oriented bounding boxes are classified into two kinds of voxels. The yellow pieces represent free voxels, and the red pieces represent occupied voxels

represents the approximate volume of the part. For an accurate volume, the algorithm can improve the resolution of the voxels. However, the higher the resolution, the higher the algorithm time complexity. We will discuss this issue in the experimental part.

4.2 | Disassembling efficiency

As to which joint is selected to disassemble, the most critical task is to minimize the wasted volume. While the parts of a group are packed into a container, the free volume is the wasted volume. The best *packing efficiency* is where the free volume is 0, which means that the volume of parts of a group is equal to the MBB[16] volume. A natural way to define the *packing efficiency* $E(g)$ of a group g is as follows:

$$E(g) = \text{Vol}(g)/\text{Vol}(\text{MBB}(g)), \tag{1}$$

where Vol(.) denotes the volume and MBB denotes the minimum-volume bounding boxes. According to Definition 1, $E(g)$ is 1; it means that the G with the best *packing efficiency* is an actual box, whereas it is smaller in all the other cases. A calculation of the group g wasted volume is helpful in the selection of the joint, and it is defined as follows:

$$W(g) = \text{Vol}(\text{MBB}(g)) - \text{Vol}(g). \tag{2}$$

According to Definition 2, the best *packing efficiency* means the wasted volume $W(g)$ is 0. The *packing efficiency* and the wasted volume of the part is the same as Definition 1 and Definition 2.

For an efficient calculation of the volume, our algorithm voxelizes the MBBs of the mechanical object. The number of voxels the mechanical group parts occupied represents the group volume Vol(g). The total number of voxels of MBBs consists of occupied voxels and free voxels.

4.3 | Disassembling joint with maximum cost

The mechanical object is treated as a collection of rigid bodies interconnected to transmit rigid motions, and a joint connects two parts to form a kinematic pair. Different joint types impose distinct motion constraints between two parts. As pointed out in the works of Mitra et al.[5] and Xu et al.,[9] the relative motion of two parts connected via a joint should be a slippable motion that does not lead to the penetration of parts, and it can be determined by the type of their intersection. Figure 1 summarizes the four main types[11] of joints used in our approach: fixed joints, revolute joints, gear-to-gear contact joints, and point-on-line joints. As shown in the work of Zhu et al.,[10] the motion of a mechanism is initialized at the driving part and transferred to other parts through its kinematic chain. In a kinematic chain, the motion of a part is restricted to the connected part based on joint type. Moreover, it is the summation of transferred motion and *own motion*. For a mechanical part, transferred motion is constant, and *own motion* is freedom without the restriction of connected parts. In our paper, we categorize the *own motion* types based on joint types as follows.

1. The two parts connected via a fixed joint move in the same direction. The motion of the part is equal to transferred motion, and *own motion* is null.
2. The gear-to-gear joint can transfer the rotational motion from one gear to another. The rotation of the part's motion is equal to the inverted rotation of transferred motion, and *own motion* is null.

3. The relative motion of a revolute joint is rotation. The translation of two axes is equal to the corresponding translation of transferred motion. The rotation and translation on one axis is *own motion*.
4. The sliding (point-on-line) joint indicates that the block can slide on the planar surface of another. The translation of one axis is equal to the corresponding translation of transferred motion. The translation of one axis and rotation is *own motion*.

Summing up the above four types, the slippable motion of a part p is considered as the summation of translation and rotation.

In a group of mechanical parts, our algorithms search the motion parameters of every part to minimize the group volume $\mathrm{vol}(g)$. The optimization function of group volume $\mathrm{vol}(g)$ can be expressed as

$$\mathrm{vol}(G) = \sum_{i=1}^{N_g} \mathrm{vol}\left(\mathrm{Rot}\left(\theta_{p_i}\right) + \mathrm{Trans}\left(\tau_{p_i}\right)\right) + \lambda \mathrm{Vol}_{\mathrm{collision}}, \tag{3}$$

where $\mathrm{Rot}(\theta_{p_i})$ and $\mathrm{Trans}(\tau_{p_i})$ are the translation and rotation of the motion parameters of a part, λ is a constant, and $\mathrm{Vol}_{\mathrm{collision}}$ is the penalty factor. Since packing with best efficiency means minimizing the group volume, we minimize Equation (3) with gradient descent techniques, where joint types are constant and motion parameters are continuous random variables. Furthermore, due to the considered running time, gradient descent techniques are used, but it is easy to fall into the local optimal solution. In order to avoid a local optimal solution, the algorithm optimizes the function with multiple initial motion parameters. The motion of two parts in one group cannot lead to the collision of parts. Thus, the penalty factor is the collision volume of two parts, and it is used to restrict the motion parameters.

The penalty factor $\mathrm{Vol}_{\mathrm{collision}}$ requires fast and robust 3D collision detection algorithms. The computational cost of a collision detection algorithm depends not only on the complexity of the basic interference test used but also on the number of times this test is applied. Some strategies in the works of Jiménez et al.,[18] Ericson,[19] and Gottschalk[20] rely on distance computation algorithms, hierarchical object representations, orientation-based pruning criteria, and space partitioning schemes. Our strategy uses collision detection based on voxelization. At the beginning of our algorithm, each part has been voxelized into several voxels. If two parts have collision, there must be overlapping voxels, and the number of overlapping voxels represents the collision volume. Using the voxelization collision detection algorithm greatly improves the efficiency of the algorithm. Therefore, we use overlapping voxels as $\mathrm{Vol}_{\mathrm{collision}}$ to optimize Equation (4).

In this section, our target is to obtain a group set with minimum total volume, which is indicated as $\mathrm{vol}(G) = \sum_{i=1}^{N_g} \mathrm{vol}(g_i)$. Initially, the mechanism is a whole object with group set G_0 and joint set C_0, where the subscript index is represented as the step of disassembly. When disassembling to the ith, a joint in joint set C_i is selected to be disconnected at the next step for minimizing the volume $\mathrm{vol}(G_{i+1})$. That means every disassembling step is maximizing reduced volume $\mathrm{vol}(G_i) - \mathrm{vol}(G_{i+1})$. Our algorithms disconnect each joint in C_i to find the maximum reduced volume.

5 | MECHANICAL GROUP PACKING

After the previous section, a mechanical object is then disassembled to a group set; our objective is to determine a roto-translation for each group so that their overall MBB is minimized. To solve this problem, our approach initializes an axis-aligned box that can contain all objects in the group set. Prior to computing such a box, however, the object is rotated according to its MBB, so that the initial box is both axis aligned and of minimum volume. That means the bottom of the box is constant and the vertical direction of the box is variant, which decides the volume size. Then, groups of parts are inserted into the box one by one while measuring their total height. Minimizing the total height is equal to minimizing the total volume. Then, optimize each part motion parameter of every group to minimize the hole volume in the box. After determining the target, the algorithm needs to solve the following three problems: the group insertion order, the group placement strategy, and motion parameter optimization.

5.1 | Insertion order

The fact that shape packing is an NP-complete problem thus justifies that heuristics and approximated solutions with assessed efficiency should be used. If the group insertion order is random, the run time will grow exponentially with the group size. Meanwhile, a difference with respect to the optimal solution is up to 70%.[13] If groups are inserted from largest to smallest, such a difference decreases down to 22%.[21] Thus, before inserting the group into the box, our algorithm sorts the groups based on their maximum extension, which, for each group, corresponds to the maximum length of its vol(MBB(g)). For mechanical object packing, first inserting the larger group is conducive to filling the holes, which is the small volume in the larger group, to improve efficiency.

5.2 | Placement of one group

Inserting a group set G into the box is to determine a roto-translation of g, as in the work of Attene.[2] Our algorithm uses quaternion Q and a vector V to define the rotation and translation of the group, respectively. Simplify the problem of optimal Q and V, and compute V for a given fixed value of Q.

Because the bottom of the box is constant, we need to find a strategy to minimize the height of the box when inserting all the mechanical groups. On the basis of the insertion order, the groups are inserted into the box one by one. The placement of each group requires an algorithmic decision to increase the space utilization of the box. In order to minimize the height of the box, the algorithm minimizes the holes as much as possible and tiles each group in the box. The algorithm proceeds as follows as depicted in Figure 4.

1. If there are holes that can contain P, we select the one whose volume is closer to P's volume and set V accordingly.
2. If no such hole exists, we set V so that P's underlying free volume is minimized while not increasing h.
3. If all the positions increase h, we set V so that h increases as few as possible.

At any stage of the algorithm, free voxels of the box can be clustered into regions of two types we call holes and slots.[2] Block a belongs to holes, and block b belongs to slots. When a new group will be inserted into the box, according to how the algorithm proceeds, we first try to place it in the hole. If this is not successful, we try to place it in the slot. Otherwise, we place it on top of the other groups in the box.

When a new group P is placed into a hole in the box, the algorithm computes the position V and the orientation Q. If the new P does not intersect any previously placed part, the V and the Q are a valid roto-translation for P. That is to say, the hole can contain P. To do that, the voxels belonging to the newly placed P are all free. First, with the orientation fixed, the algorithm finds a V through the gradient descent algorithm to minimize the number of collision voxels, which overlap the newly placed P and the previously placed part. Then, rotate the newly placed P until finding the valid V and Q. If this is successful, the newly placed P can be inserted into the holes. Otherwise, no such hole exists, and the algorithm will look for a suitable slot.

When inserting a newly placed P into the slot, our algorithm finds inspiration from the work of Sander et al.[22] Let P be the part to be placed into the slot, and minimize the height of P. The algorithm needs to optimize both parameters V and Q at the same time. The initial value of V is the center of the slot's OBB, and the initial value of Q is the unit matrix. If no such slot can contain a newly placed P, the algorithm inserts P on top of the box. The algorithm fixes the rotation Q and sets V to minimize the increasing height.

The packing algorithm defines a single cost to be minimized as follows:

$$\mathrm{Cost}(P) = \Delta_h B + U, \tag{4}$$

where Δ_h is the increase in h, U is P's underlying free volume, and B is the total volume of the box. To include Q in the cost minimization process, we simply add an outer loop where a set of predefined rotations is iteratively assigned to Q. The algorithm uses uniform sampling[23] of the space of unit quaternions to compute predefined rotations.

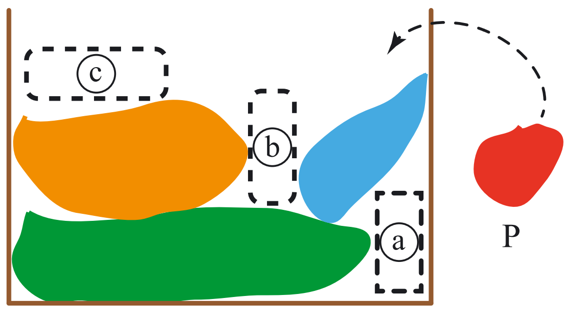

FIGURE 4 Inserting a group set into the brown box. There are already three groups (yellow part, blue part, and green part) in the box. There are three places where the red P can be placed, and they are the dotted a, dotted b, and dotted c, respectively

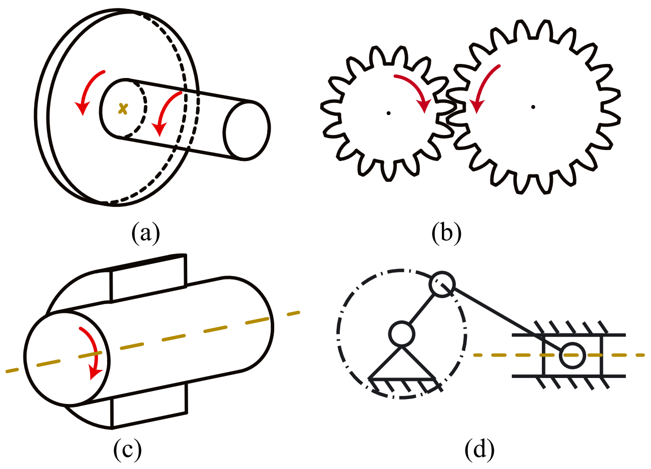

FIGURE 5 Joint types connecting two parts. (a) Fixed joint. (b) Gear-to-gear joint. (c) Revolute joint. (d) Sliding joint

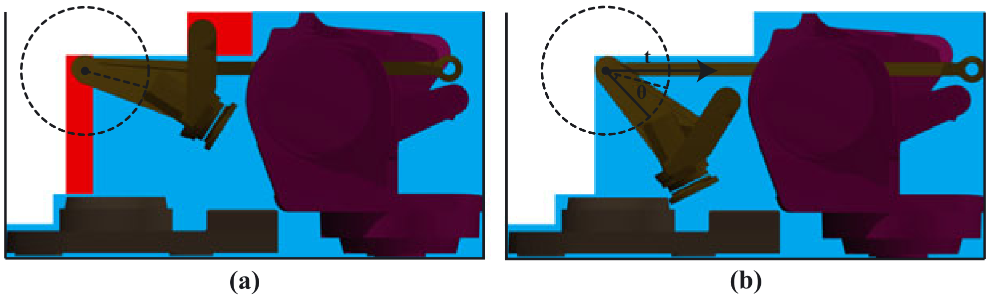

FIGURE 6 Motion parameter optimization. The blue area is the groups placed into the box. When a new mechanical group is placed into the box, the translation t and rotation θ of each part should be optimized to minimize the hole volume, as shown on the right. After optimization, the red area is the space saved. (a) Before optimization. (b) After optimization

5.3 | Motion parameter optimization

The above packing algorithm is similar to the work of Attene.[2] However, for the mechanical model packing problem, this algorithm can only get the approximate position in the box. We can improve the space utilization by optimizing each part's motion parameter of every packed group. As mentioned above, minimizing the volume of the hole in the box is equivalent to increasing the space utilization. Hence, when we place each group into the box, our goal is to reduce the volume of the hole. Instead of a static model, the slippable motion of each part of the mechanism can be adjusted to match the shape of the hole in the box. The relative motion of two parts connected via a joint based on the type of their intersection has been presented in Section 4.3 and shown in Figure 5. Similar to the previous optimization, our algorithm optimizes each part's motion parameter to minimize the hole volume when the group is placed into the box. As shown in Figure 6, utilizing the advantages of the mechanism joint freedom, the motion parameter of each part of the placed group is optimized, making the group part more compact and, thus, improving packing space utilization.

5.4 | Overall packing algorithm

To summarize our algorithm, the initial input is a mechanical model, and the initial parameters are the space utilization $U_{\min}$, the bottom size of the box, and the maximum number of disassembly $S_{\max}$. The goal of our algorithm is to acquire maximum space utilization with minimum number of disassembly when placing a mechanical model into the box through Algorithm 1.

Algorithm 1. Overall disassembly-and-packing algorithm

Input: The mechanical model M; The minispace utilization $U_{\min}$; maximum number of disassembly $S_{\max}$;
Output: A disassembled Group set G; Each group with its rotation Q and translation V;

```
1:  initial $U_0 = 0$ and $S_0 = 0$;
2:  while $U_k < U_{\min}$ do
3:      Disassemble the Mechanism base on a selected connection to generate new group set $G_k$;
4:      Update the $S_k$;
5:      if $S_k > S_{\min}$ then
6:          Terminate with failure;
7:      else
8:          Create a queue $M$ for storing sorted group;
9:          for all $P \in G$ do
10:             Based on joint type, rotate and translate each part of $P_i$;
11:             Compute the OBB($P_i$);
12:             Insert $g_i$ into the descending queue $M$;
13:         end for
14:         for all $P \in M$ do
15:             if There are holes that can contain $P_i$ then Actually roto-translate $P_i$ using the $q$ and $v$ that minimized the wasted space;
16:             else
17:                 if There are slots that can contain $P_i$ then Actually roto-translate $P_i$ using the $q$ and $v$ that minimized the height of the parts;
18:                 else
19:                     Put the part $P_i$ on the top of the box that minimized the height of the parts. And compute the $q$ and $v$ of the $P_i$'s roto-translate;
20:                 end if
21:             end if
22:             Optimize the motion parameter of each part to minimize the hole volume of the box.
23:         end for
24:     end if
25: end while
26: Return $G$, $Q$ and $V$;
```

6 | RESULTS AND DISCUSSION

We have implemented the system on a desktop PC with an Intel I7 CPU (4.20 GHz) and 16-GB memory. As shown in Figures 1 and 7, our experiments selected four representative mechanical objects, namely, robot-arm, crank-block, motor, and excavator, which are disassembled and packed into a box. In Section 5, the initial bounding box is discretized using 256^3 voxels, which is implemented to calculate the volume of the parts at the same scale. Detailed statistics of the packing mechanical object are reported in Table 1.

In order to improve the packing efficiency, more groups of the model disassembled is profitable. Unfortunately, for the mechanical object, more joints are connected, resulting in increased mechanical part disassembly time and the mechanism being more likely to be destroyed. In our experiments, the total number of groups in Table 1 and the packing result in Figure 7 show that the mechanical object is disassembled into a few groups, which retains the main structure of the mechanism. The data show that more than two thirds of the joints are still connected.

In the mechanical group packing experiment, the initial bottom of the box influences the eventual packing efficiency. Before inserting the group set into the box, the maximum width and length of the parts have been calculated in Section 5.1. Our experiment took four scales to set the initial bottom size, which are 1.0, 1.25, 1.5, and 2 times of maximum width and length. A smaller bottom size setting is adverse to the case with many large parts. Meanwhile, in the case with a larger bottom size setting, there are many holes. As a result, our experiment observes that the bottom size set at 1.25 time will acquire maximum packing efficiency.

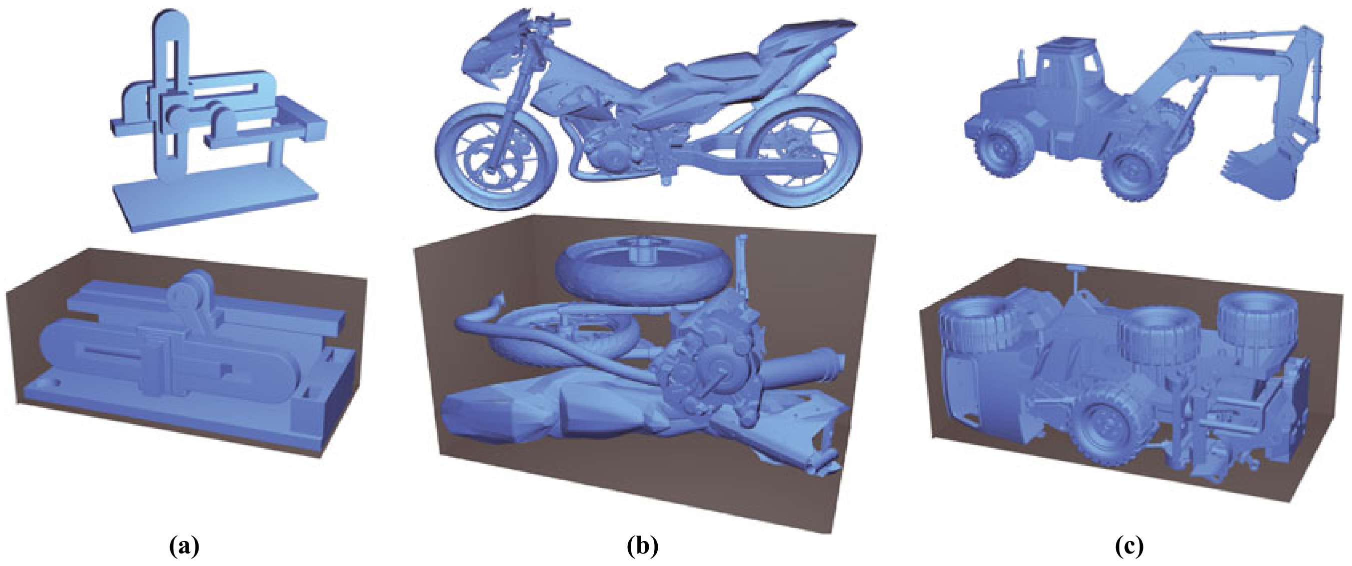
FIGURE 7 Some example models used to test our algorithm. (a) Crank-block. (b) Motor. (c) Excavator

TABLE 1 Statistics of the packing mechanical object. #Parts, #Joints, and #Groups denote the total number of parts, joints, and groups, respectively

Name	#Parts	#Joints	Init efficiency	#Groups	Pack efficiency	Running time (s)
Robot-arm	18	21	9.4%	5	27.2%	9
Crank-block	9	9	13.6%	5	38.0%	6
Motor	29	28	5.8%	6	17.2%	22
Excavator	32	37	6.2%	11	15.4%	31

TABLE 2 Packing efficiency result

Name	splitPack	Ours
Robot-arm	13.2%	27.2%
Crank-block	23.9%	38.0%
Motor	8.6%	17.2%
Excavator	9.1%	15.4%

While this paper was under review, a quite similar work was published by another group.[2] The concurrent algorithm, called splitPack, aims at splitting an object and packing the resulting parts. Different from the splitPack, our mechanism object has an amount of joints and parts, and quite a few joints are unnecessary to be split. The disassembled group can improve space utilization by rotating and translating the parts, which are subjected to the constraints of joint type. In order to facilitate the assembly of mechanical models, the algorithm aims at minimizing the number of disassembly with satisfied packing efficiency. At the same number of disassembly, compared with splitPack, our algorithm has a significant improvement in space utilization (see Table 2).

7 | DISCUSSION AND CONCLUSION

In this paper, we have presented a packing algorithm to disassemble a mechanism into a group set and insert them into a box. It consists of two main steps: The first step is disassembling the joint with maximum cost and adjusting each part in the group with maximum efficiency. The second step is sorting the group by volume and inserting the group into the box. The above two steps are repeated until the packing efficiency satisfies the initial requirements.

Limitations and future work. Because of the running time, our approach uses the strategy of the greedy algorithm to disassemble the mechanism object. Hence, we cannot prove that the current disassembly result has the best efficiency. For a complex mechanical object, disassembly order is based on the connection topology of the parts. Disassembly order influences the eventual packing efficiency; hence, the order should be considered in our algorithm.

ACKNOWLEDGEMENTS

We would like to thank the anonymous reviewers for their constructive comments. Mingliang Xu is partially supported by National Science Foundation of China (NSFC) (No. 61672469, No. 61472370 and No. 61822701) and program for Science & Technology Innovation Talents in Universities of Henan Province (HASTIT) (No. 18HASTIT020) and program for Young Scholar in Universities of Henan Province (No. 2016GGJS-001). Enjie Ding was partially supported by the National Key Technologies R&D Program of China under Grant 2017YFC0804401. Weiwei Xu is partially supported by NSFC (No. 61732016), Alibaba IDEA Lab and fundamental research fund for the central universities (2017YFB1002600).

ORCID

Mingyuan Li http://orcid.org/0000-0002-9397-751X

REFERENCES

1. Chen X, Zhang H, Lin J, et al. Dapper: decompose-and-pack for 3D printing. ACM Trans Graph. 2015;34(6). Article No. 213.
2. Attene M. Shapes in a box: disassembling 3D objects for efficient packing and fabrication. Comput Graph Forum. 2015;34:64–76.
3. Li M, Jiang X, Gu N, et al. Efficiently disassemble-and-pack for mechanism.
4. Dong T, Zhang L, Tong R, Dong J. A hierarchical approach to disassembly sequence planning for mechanical product. Int J Adv Manuf Tech. 2006;30(5–6):507–520.
5. Mitra NJ, Yang Y-L, Yan D-M, Li W, Agrawala M. Illustrating how mechanical assemblies work. ACM Trans Graph. 2010;29(4). Article No. 58.
6. Coros S, Thomaszewski B, Noris G, et al. Computational design of mechanical characters. ACM Trans Graph. 2013;32(4). Article No. 83.
7. Xu M, Zhu J, Lv P, Zhou B, Tappen MF, Ji R. Learning-based shadow recognition and removal from monochromatic natural images. IEEE Trans Image Process. 2017;26(12):5811–5824.
8. Mingliang X, Lv P, Li M, et al. Medical image denoising by parallel non-local means. Neurocomputing. 2016;195(C):117–122.
9. Xu W, Wang J, Yin K, et al. Joint-aware manipulation of deformable models. ACM Trans Graph. 2009;28. Article No. 35.
10. Zhu L, Xu W, Snyder J, Liu Y, Wang G, Guo B, Motion-guided mechanical toy modeling. ACM Trans Graph. 2012;31(6). Article No. 127.
11. Xu M, Li M, Xu W, Deng Z, Yang Y, Zhou K. Interactive mechanism modeling from multi-view images. ACM Trans Graph. 2016;35(6). Article No. 236.
12. Xu M-L, Gu N-B, Xu W-W, Li M-Y, Xue J-X, Zhou B. Mechanical assembly packing problem using joint constraints. J Comput Sci Technol. 2017;32(6):1162–1171.
13. Garey MR, Graham RL, Ullman JD. An analysis of some packing algorithms. Combinatorial Algorithms; 1973:39–47.
14. Chernov N, Stoyan Y, Romanova T. Mathematical model and efficient algorithms for object packing problem. Comput Geom. 2010;43(5):535–553.
15. Bansal N, Han X, Iwama K, Sviridenko M, Zhang G. A harmonic algorithm for the 3D strip packing problem. SIAM J Comput. 2013;42(2):579–592.
16. O'Rourke J. Finding minimal enclosing boxes. Int J Parallel Programming. 1985;14(3):183–199.
17. Attene M, Campen M, Kobbelt L. Polygon mesh repairing: an application perspective. ACM Comput Surv. 2013;45(2). Article No. 15.
18. Jiménez P, Thomas F, Torras C. 3D collision detection: a survey. Comput Graph. 2001;25(2):269–285.
19. Ericson C. Real-time collision detection. Boca Raton, FL: CRC Press, Inc.; 2004.
20. Gottschalk SA. Collision queries using oriented bounding boxes [PhD dissertation]. Chapel Hill, NC: The University of North Carolina at Chapel Hill; 2000.
21. Johnson DS. Approximation algorithms for combinatorial problems. Proceedings of the Fifth Annual ACM Symposium on Theory of Computing; 1973 May 2; Austin, TX. New York, NY: Association for Computing Machinery; 1973. p. 38–49.
22. Sander PV, Wood ZJ, Gortler SJ, Snyder J, Hoppe H. Multi-chart geometry images. Proceedings of the 2003 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing; 2003 Jun 23–25; Aachen, Germany. Aire-la-Ville, Switzerland: Eurographics Association; 2003.
23. Marsaglia G. Choosing a point from the surface of a sphere. Ann Math Stat. 1972;43(2):645–646.

Mingyuan Li is a Ph.D. candidate in the School of Information Engineering, Zhengzhou University, Zhengzhou, China. His research interest is in computer graphics, specifically for 3D reconstruction, motion capture, and optimization.

Xiaoheng Jiang received the B.S., M.S., and Ph.D. degrees in Electronic Information Engineering from Tianjin University, Tianjin, China, in 2010, 2013, and 2017, respectively. Currently, he is a Lecturer in the School of Information Engineering, Zhengzhou University, Zhengzhou, China. His research interests include computer vision and deep learning.

Ningbo Gu received the M.S. degree from the School of Information Engineering, Zhengzhou University, Zhengzhou, China. His research interest is in computer graphics and computer vision.

Weiwei Xu received the B.S. and Master's degrees in Computer Science from Hohai University, Nanjing, China, in 1996 and 1999, respectively, and the Ph.D. degree in Computer Graphics from Zhejiang University, Hangzhou, China. Currently, he is a Researcher in the Department of Computer Science, Zhejiang University. Before that, he was a Qianjiang professor at Hangzhou Normal University and a researcher in the Internet Graphics Group, Microsoft Research Asia, from October 2005 to June 2012. He was a postdoctoral researcher at Ritsumeikan University, Kyoto, Japan, around one year from 2004 to 2005. He has published more than 60 papers in international conference and journals, including 16 papers on ACM Transactions on Graphics. His main research interests include digital geometry processing and physics-based simulation techniques. He is now focusing on 3D reconstruction and computational fabrication techniques.

Junxiao Xue received the Ph.D. degree from the School of Mathematical Sciences, Dalian University of Technology, Dalian, China, in 2009. Currently, he is an associate professor in the School of Software, Zhengzhou University, Zhengzhou, China. His research interests include computer graphics and computer-aided geometric design.

Bing Zhou received the B.S. and M.S. degrees from Xi'an Jiaotong University, Xi'an, China, in 1986 and 1989, respectively, and the Ph.D. degree from Beihang University, Beijing, China, in 2003, all in computer science. Currently, he is a professor in the School of Information Engineering, Zhengzhou University, Zhengzhou, China. His research interests cover video processing and understanding, surveillance, computer vision, and multimedia applications.

Mingliang Xu received the Ph.D. degree in computer science and technology from the State Key Laboratory of CAD&CG, Zhejiang University, Hangzhou, China. Currently, he is a full professor in the School of Information Engineering, Zhengzhou University, Zhengzhou, China. He is also the Director of the Center for Interdisciplinary Information Science Research and the Vice General Secretary of ACM SIGAI China. His current research interests include computer graphics, multimedia, and artificial intelligence.

How to cite this article: Li M, Jiang X, Gu N, et al. Disassembling a 3D mechanism for efficient packing. *Comput Anim Virtual Worlds*. 2018;e1861. https://doi.org/10.1002/cav.1861