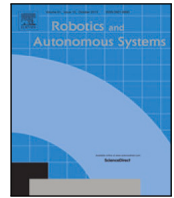| Title	Many-objective-optimized semi-automated robotic disassembly sequences
Author(s)	Kiyokawa, Takuya; Harada, Kensuke; Wan, Weiwei et al.
Citation	Robotics and Autonomous Systems. 2025, 197, p. 105301
Version Type	VoR
URL	https://hdl.handle.net/11094/103704
rights	This article is licensed under a Creative Commons Attribution 4.0 International License.
Note	

Many-objective-optimized semi-automated robotic disassembly sequences☆

Takuya Kiyokawa [a] [iD],*,[1], Kensuke Harada [a,b],[2], Weiwei Wan [a],[2], Tomoki Ishikura [c], Naoya Miyaji [c], Genichiro Matsuda [c]

[a] Department of Systems Innovation, Graduate School of Engineering Science, The University of Osaka, 1-3 Machikaneyama-cho, Toyonaka-shi, Osaka, Japan
[b] Industrial Cyber-physical Systems Research Center, The National Institute of Advanced Industrial Science and Technology (AIST), 2-3-26 Aomi, Koto-ku, Tokyo, Japan
[c] Manufacturing Innovation Division, Panasonic Holdings Corporation, 2-7 Matsuba-cho, Kadoma, Osaka, Japan

ARTICLE INFO

ABSTRACT

This study tackles the problem of many-objective sequence optimization for semi-automated robotic disassembly operations. To this end, we employ a many-objective genetic algorithm (MaOGA) inspired by the non-dominated sorting genetic algorithm (NSGA)-III, along with robotic-disassembly-oriented constraints and objective functions derived from geometrical and robot simulations using three-dimensional (3D) geometrical information stored in a 3D computer-aided design (CAD) model of the target product. The MaOGA begins by generating a set of initial chromosomes based on a contact and connection graph (CCG), rather than random chromosomes, to avoid falling into a local minimum and yield repeatable convergence. The optimization imposes constraints on feasibility and stability as well as objective functions regarding difficulty, efficiency, prioritization, and allocability to generate a sequence that satisfies many preferred conditions under mandatory requirements for semi-automated robotic disassembly. The NSGA-III-inspired MaOGA also utilizes non-dominated sorting and niching with reference lines to further encourage steady and stable exploration and uniformly lower the overall evaluation values. Our sequence generation experiments for a complex product (36 parts) demonstrated that the proposed method can consistently produce feasible and stable sequences with a 100% success rate, bringing the multiple preferred conditions closer to the optimal solution required for semi-automated robotic disassembly operations.

1. Introduction

With the aim of achieving a sustainable society, there has been greater emphasis on promoting recycling, reuse, and remanufacturing. In particular, future manufacturing robots are expected to exhibit proficiency in efficient disassembly of many parts. In this context, autonomous robotic disassembly has garnered increased attention [1, 2]. The remanufacturing domain requires the deployment of robots capable of autonomously acquiring sequential disassembly operations in an efficient and streamlined manner to address a diverse array of needs. Furthermore, human–robot cooperation has been promising for carefully extracting valuable parts from disassembly target products [3].

To achieve the sequential disassembly operations without much manual effort, the automatic generation of sequences is crucial. To automatically generate sequences for (dis)assembly, previous studies have employed a three-dimensional (3D) model of the product [4–7]. Determining the order of (dis)assembly parts can be categorized as a combinatorial optimization problem and a non-deterministic polynomial-time (NP)-hard problem [8], which necessitates the use of heuristic search algorithms to obtain a sub-optimal solution within a practical timeframe.

To facilitate robotic disassembly for flexible remanufacturing effectively, it is crucial to consider various aspects when determining the required sequence. Several studies have explored disassembly sequence planning (DSP) in the context of semi-automated processes
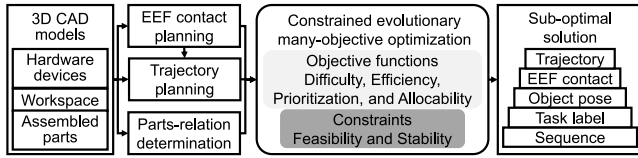
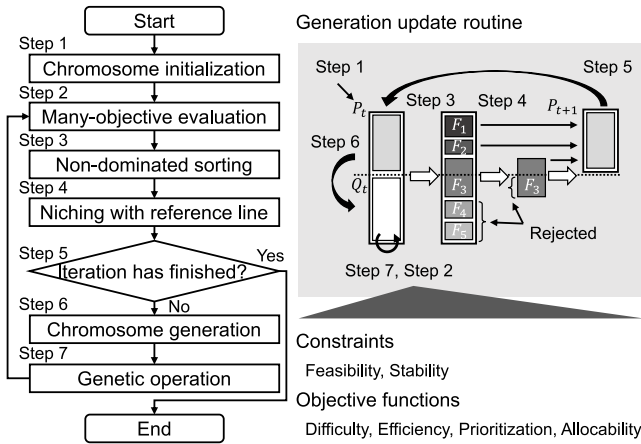**Fig. 1.** Overview of robotic disassembly planning.

Fig. 2. NSGA-III-inspired sequence optimization.

or human–robot-collaboration (HRC) by considering different perspectives [3,9–14]. To generate a disassembly sequence, they evaluated the disassembly cost, disassemblability, and safety from the long-horizon perspective of the sequence to optimize the overall order of parts and processes in real-time.

However, they did not address many-objective optimization problems (*i.e.*, involving four or more objectives) under constraints and objective functions specific to semi-automated robotic disassembly with the planning of robot operations, as in our study. Therefore, this study addresses robotic DSP rather than merely optimizing the preferable parts order and desirable processes.

Fig. 1 shows the overview of our assumed robotic disassembly planning. Specifically, this study establishes constraints on disassembly order feasibility, robot motion feasibility, and object placement stability. The designed objective functions pertain to order difficulty based on the contact state transition difficulty, task efficiency determined by the number of end-effector (EEF) changes and distance between adjacent pairs in the sequence, prioritization based on the user-defined priority of disassembling particular parts before others, and allocability based on the number of task-allocated agent changes between humans and robots.

To address the many-objective optimization problem with conflicting objectives, we employ a many-objective genetic algorithm (MaOGA) inspired by non-dominated sorting genetic algorithm III (NSGA-III) [15], a state-of-the-art evolutionary many-objective optimization algorithm. Fig. 2 shows an overview of the proposed algorithm. Generally, to avoid convergence to local optimal solutions, heuristic search algorithms need to effectively generate as many constraint-satisfied initial solutions as possible. To generate chromosomes represented as a set of sequences, we propose to generate a contact and connection graph (CCG) representing the contact and connection relationships among disassembly parts and use the CCG to efficiently generate constraint-satisfied initial solutions. The generation of the disassembly order of parts can be achieved through simple
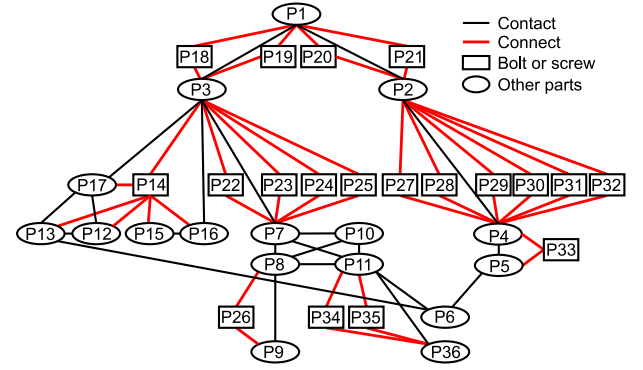
(a) Contact and connection graph (CCG).
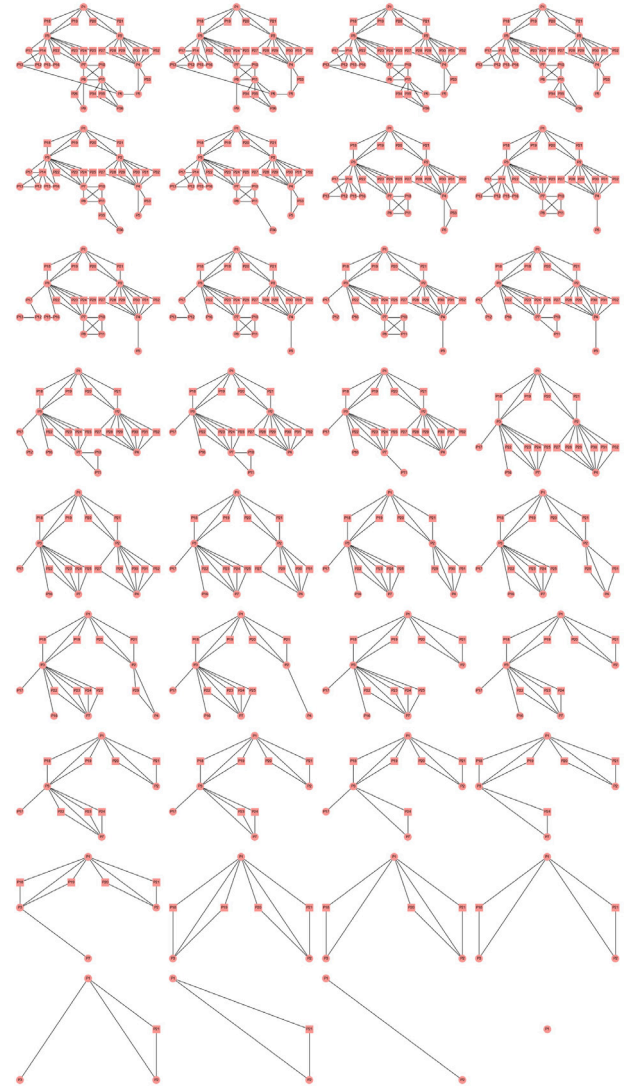
(b) An example of the generated initial solution.

Fig. 3. CCG-based initialization.

step-by-step removal of the end nodes of the graph. Fig. 3 shows the overview of CCG-based initialization (CCGI). Fig. 3(a) shows an example of the CCG for the target product used for our experiments in this study.

This study examines the effectiveness of the proposed stability-based initial chromosome generation method depicted in Fig. 3(b) by verifying that it outperforms random chromosome initialization. The NSGA-III-inspired algorithm also utilizes non-dominated sorting and niching with reference lines to further encourage steady and stable exploration of the solutions and uniformly lower the overall evaluation values. Our simulation experiment verifies the applicability of the newly-constructed NSGA-III-inspired algorithm tailored for the robotic DSP problem.

In the evaluation, we show that chromosome initialization repeatedly generates the most interference-free and stable initial solutions by comparing with other initialization methods. Our ablation study further show that the NSGA-III-inspired algorithm can steadily and effectively reduce the evaluation values through many-objective optimization based on non-dominated sorting and niching with reference lines. Finally, the proposed algorithm successfully generated disassembly sequences and robotic disassembly operations for a complex belt drive unit composed of 36 parts while considering multiple necessary constraints and desirable objectives from diverse perspectives, including semi-automated robotic operations.

In the rest of this paper, we discuss related work in Section 2, describe our proposed algorithm in Section 3, present the evaluation results in Section 4, discuss the remaining issues in Section 5, and conclude the paper in Section 6.

2. Related work

2.1. Determining order of parts

The optimization and determination of the order of parts in (dis)assembly processes have been the focus of extensive research for several decades [5,11,14,16–25]. The growing demand for high-mix, low-volume production and significant advancements in computer capabilities have recently revived interest in this topic in the field.

Kiyokawa et al. [21] proposed a multi-objective genetic algorithm specifically tailored for assembly sequence planning (ASP). The experimental outcome suggests that the proposed NSGA-II [26]-inspired multi-objective genetic algorithm can identify Pareto optimal solutions across a range of objective functions. NSGA-II generally employs non-dominated sorting, which ranks solutions based on Pareto dominance to approximate the optimal front, and crowding distance to maintain diversity. However, since crowding distance is less effective in high-dimensional spaces, our study employs NSGA-III, which is an extension of this framework. NSGA-III replaces the crowding distance with niching based on reference lines, where individuals are associated with predefined reference points on a hyperplane to ensure a uniform distribution of solutions in many-objective optimization problems.

Ebinger et al. [27] introduced a flexible DSP framework which includes a subassembly identification method. The authors investigated the usefulness of subassemblies in search by examining the framework performance with and without subassemblies. Chervinskii et al. [6] introduced Auto-Assembly, a framework that encompasses design analysis, ASP, Bill-of-Process (BOP) generation, and control code execution for physical assembly. Dorn et al. [23] addressed the challenge of generating a Voronoi diagram for a complex automotive model, and developed an assembly priority graph. Wang et al. [24] focused on robotic parallel DSP for end-of-life products and proposed a multi-objective model to minimize the makespan and energy consumption.

Several studies have solved several cases of path-planning problems for ASP and DSP. Lee et al. [28] achieved minimization of the steps to goal component removal based on the concept of blocking topology. Le et al. [19] extended a sampling-based path planner Rapidly-exploring Random Tree (RRT) for ASP and DSP for LEGO bricks. Moreover, Tian et al. [25] proposed a physics-based assembly-by-disassembly planner using a large-scale dataset, achieving state-of-the-art performance.

However, these studies [6,19,21,23–25,27,28] did not address the finer aspects of robot motions, such as EEF's contacts and trajectories, which are critical for robotic disassembly operations.

Recent studies have employed graph neural networks to deduce a feasible sequence by analyzing the graph representation of the complex (dis)assembly model structure. Cebulla et al. [29] introduced an assembly-by-disassembly approach that involves iteratively testing parts for removal, with the testing order significantly impacting runtime. The authors optimized the order using a graph neural network trained on part shapes and local connections. Ma et al. [30] introduced a heterogeneous graph-transformer framework for learning the latent rules of assembly planning. However, these studies [29,30] did not address robotic disassembly and limited the assembly targets to aluminum profiles or LEGO bricks consisting of a small number of parts compared to actual mechanical products. Recent studies on multi-objective optimization of disassembly orders [10,13,20,31] have focused on improving heuristic algorithms, rather than establishing an optimization framework that considers robotic disassembly operations.

2.2. Robotic assembly and disassembly

On the other hand, the process of planning (dis)assembly robot motions necessitates taking into account constraints in multiple dimensions and perspectives. Thus far, several researchers have successfully generated and executed disassembly operations using robots, assuming that the (dis)assembly order is given or that the operations involve two separate parts [32–35].

Rodriguez et al. [36] proposed iteratively checking multilevel feasibilities to plan assembly sequences with robot motions. Bachmann et al. [37] investigated the impact of robotic workcell layout on task efficiency and feasibility. Most recently, Atad et al. [38] used a graph neural network to infer feasible and optimal assembly sequences by learning an inference model based on a geometric structure graph representation of a product. However, these studies [36–38] more focused on the aluminum profile assemblies consisting of a small number of parts compared to mechanical products.

Liu et al. [39] succeeded in optimizing the sequence and process of robotic disassembly through collaborative optimization; however, they did not take into account feasible motions (i.e., EEF's contacts and trajectories). Koga et al. [7] proposed a computer-aided design (CAD)-based robotic assembly system; however, generating sequential and semi-automated disassembly operations optimized using multiple objective functions remains an open issue.

Laili et al. [22] addressed the issue of flexible sequencing in robotic disassembly with failed automation operations, proposing an online recovery method that utilizes pre-stored backup actions. Jiayi et al. [40] used a digital twin for dynamic planning of robotic disassembly under uncertain real-world conditions. These studies [22,40] focused on different issues from our study. Learning-based assembly planning approaches [4,41–44] have demonstrated promising results in adapting to a broader range of products; however, their efficacy has primarily been validated with toy objects or furniture. Unlike these studies, this study does not involve the exploration of dynamic planning and learning-based generalization.

This study does not delve into other issues related to disassembly specific system components, including EEFs, controllers, and interfaces, as reported in [9,12,45–48]. Gorjup et al. [49] introduced an integrated flexible manufacturing system that uses compliance control, CAD-based localization, and multimodal gripper for efficient assembly task programming. However, this system was limited to part-kitting tasks only.

In contrast, our study aims to explore the feasibility of generating effective disassembly sequences for a complex product by designing suitable constraints, objective functions, chromosome initialization rules, and genetic generation update rules in the proposed method.

Algorithm 1 Robotic DSP

Input: 3D CAD models of parts and environment $\boldsymbol{M}$
Output: An optimum order $\hat{\boldsymbol{O}}$, disassembly task label $\hat{\boldsymbol{L}}$, EEF contact
　$\hat{\boldsymbol{C}}$, arm trajectory $\hat{\boldsymbol{T}}$, and object pose $\hat{\boldsymbol{P}}$
1: $\boldsymbol{I}, \boldsymbol{X}^{if}, \boldsymbol{X}^{cs}, \boldsymbol{X}^{ct}, \boldsymbol{X}^{cf}$ = GeometricalSimulation($\boldsymbol{M}$)
2: $\boldsymbol{X}^{mf}, \boldsymbol{C}, \boldsymbol{T}, \boldsymbol{P}$ = RobotSimulation($\boldsymbol{M}$)
3: Set $T^i_{\max}, T^g_{\max}, \boldsymbol{P}^{ga}$ with user inputs
4: SetPlanner($\boldsymbol{I}, \boldsymbol{X}^{if}, \boldsymbol{X}^{cs}, \boldsymbol{X}^{ct}, \boldsymbol{X}^{cf}, \boldsymbol{X}^{mf}, \boldsymbol{C}, \boldsymbol{T}, \boldsymbol{P}, \boldsymbol{P}^{ga}$)
5: Initialize the counters of t^i and t^g to 1
6: **while** $t^i < T^i_{\max}$ **do**
7: 　$\boldsymbol{G}^1$ = ChromosomeInitialization()
8: 　**while** true **do**
9: 　　Initialize the elements of $\boldsymbol{E}^{G^{t^g}}$ to 1
10: 　　$\boldsymbol{C}^{G^{t^g}}$ = ConstraintCheck($\boldsymbol{G}^{t^g}$)
11: 　　**if** All elements of $\boldsymbol{C}^{G^{t^g}}$ are *Available* **then**
12: 　　　$\boldsymbol{E}^{G^{t^g}}$ = FitnessCalculation($\boldsymbol{G}^{t^g}$)
13: 　　$\boldsymbol{G}^{best}$ = BestSolutionExtraction($\boldsymbol{G}^{t^g}$, $\boldsymbol{E}^{G^{t^g}}$)
14: 　　**if** $T^g_{\max} < t^g$ **then**
15: 　　　**break**
16: 　　$\tilde{\boldsymbol{G}}^{t^g}, \boldsymbol{E}^{\tilde{G}^{t^g}}$ = NonDominatedSorting($\boldsymbol{G}^{t^g}, \boldsymbol{E}^{G^{t^g}}$)
17: 　　NichingWithReferenceLine($\tilde{\boldsymbol{G}}^{t^g}, \boldsymbol{E}^{\tilde{G}^{t^g}}$)
18: 　　$\boldsymbol{G}^{t^g+1}$ = NextGenerationCreation($\tilde{\boldsymbol{G}}^{t^g}, \boldsymbol{E}^{\tilde{G}^{t^g}}$)
19: 　　$\boldsymbol{G}^{t^g+1}$ = GeneticOperation($\boldsymbol{G}^{t^g+1}$)
20: 　　$t^g = t^g + 1$
21: 　$t^i = t^i + 1$
22: $\hat{\boldsymbol{O}} = \boldsymbol{G}^{best}$
23: $\hat{\boldsymbol{L}}$ = LabelExtraction($\hat{\boldsymbol{O}}$)
24: $\hat{\boldsymbol{C}}, \hat{\boldsymbol{T}}, \hat{\boldsymbol{P}}$ = OperationParameterExtraction($\hat{\boldsymbol{O}}$)

---

3. Optimizing robotic disassembly sequence

3.1. Overview

Algorithm 1 outlines the flow of the robotic DSP algorithm. In our algorithm, the order is represented by O_k ($k = 1, \ldots, N^p$), where N^p represents the number of parts. This sequence corresponds to a chromosome in the proposed MaOGA, encoded as a permutation of integer numbers (*i.e.*, part indices). The last part is denoted by O_1, and the first part is denoted by O_{N^p}. The order obtained through optimization is denoted by $\hat{\boldsymbol{O}}$. The algorithm takes 3D CAD models of the target parts and environment (*e.g.*, a robot arm, EEF, and worktable), represented by $\boldsymbol{M}$, as inputs. The outputs of the algorithm include the optimized order $\hat{\boldsymbol{O}}$, along with the task labels $\hat{\boldsymbol{L}}$, EEF contacts $\hat{\boldsymbol{C}}$, arm trajectories $\hat{\boldsymbol{T}}$, and object placement poses $\hat{\boldsymbol{P}}$.

Our DSP process commences with an in-depth analysis of the geometries derived from CAD models $\boldsymbol{M}$, subsequently generating parts-relation matrices. The GeometricalSimulation($\boldsymbol{M}$) function first extracts the labels associated with part names, as well as information pertaining to the center of mass, pose, and shape of each part, represented by $\boldsymbol{I}$. Subsequently, it provides us with interference-free $\boldsymbol{X}^{if}$, constraint degree $\boldsymbol{X}^{cs}$, contact $\boldsymbol{X}^{ct}$, and constraint-free $\boldsymbol{X}^{cf}$ matrices, which encapsulate diverse types of relationships between each pair of parts.

The function RobotSimulation($\boldsymbol{M}$) provides the motion-feasibility matrices $\boldsymbol{X}^{mf}_i$ ($i = 1, \ldots, N^p$), feasible contacts $\boldsymbol{C}$, trajectories $\boldsymbol{T}$, and placement poses $\boldsymbol{P}$ for each part. After planning the contacts to initiate the manipulation of the target part P_i and generating a multitude of collision-free contact samples, each sample is evaluated for collision-free and Inverse Kinematics (IK)-solvable trajectories for all specified placement poses. Furthermore, we check if there exists a trajectory enabling the undisassembled target part to be moved from the placement pose to other placement poses (possible next placement pose). The dataset used in this study consists of all possible combinations of placement poses for all possible compositions of the undisassembled target

object, which has been previously examined. For both the disassembly trajectory and transportation trajectory between the two placement poses, if at least one trajectory set is generated, the corresponding binary element of $\boldsymbol{X}^{mf}_i$ can be 1 (feasible). If multiple feasible trajectory sets exist for the same placement poses, the shortest trajectory is selected to guarantee minimal trajectory length. The feasibility is stored in the corresponding element of $\boldsymbol{X}^{mf}_i$. The elements of $\boldsymbol{X}^{mf}_i$ indicate whether each feasible motion (including feasible contacts, trajectories, and object placement poses) generated for the target part P_{O_i} interferes with parts other than P_{O_i}.

The user specifies the MaOGA parameters, including the maximum number of iterations $T^i_{\max}$, the maximum number of generation updates $T^g_{\max}$, and a set of other MaOGA parameters $\boldsymbol{P}^{ga}$. $\boldsymbol{P}^{ga}$ includes the number of chromosomes, crossover rate, mutation rate, cut-and-paste rate, and break-and-join rate. The matrices and parameters are then set to the planner using the SetPlanner() function. Once the planner is set, the matrices and parameters can be accessed from any function in the algorithm.

Prior to initiating the optimization loop, the values of t^i and t^g are both set to 1. The optimization process commences with the initialization of the first-generation chromosome $\boldsymbol{G}^1$ using ChromosomeInitialization(). Subsequently, the optimization (generation update) loop commences, where genes are evaluated using ConstraintCheck($\boldsymbol{G}^{t^g}$) and FitnessCalculation($\boldsymbol{G}^{t^g}$). The best solution $\boldsymbol{G}^{best}$ is extracted using with BestSolutionExtraction($\boldsymbol{G}^{t^g}$, $\boldsymbol{E}^{G^{t^g}}$). Additionally, based on the evaluation values $\boldsymbol{E}^{G^{t^g}}$, $\boldsymbol{G}^{t^g}$ is sorted using NonDominatedSorting($\boldsymbol{G}^{t^g}, \boldsymbol{E}^{G^{t^g}}$). The sorted solutions $\tilde{\boldsymbol{G}}^{t^g}$ along with their evaluation values $\boldsymbol{E}^{\tilde{G}^{t^g}}$ are associated with the reference lines using the NichingWithReferenceLine($\tilde{\boldsymbol{G}}^{t^g}, \boldsymbol{E}^{\tilde{G}^{t^g}}$). Reference lines are created from points at infinity to the optimal point. The assignment of solutions with reference lines guides the exploration direction of each solution. By repeatedly applying this process, NSGA-III steers the exploration. Consequently, when the algorithm converges, the reference lines facilitate the discovery of a superior representative non-dominated solution.

Following the selection of solutions (sequences) to which the genetic operations are applied based on the assignment with the reference line, the process advances to the creation of the next generation $\boldsymbol{G}^{t^g+1}$ through the NextGenerationCreation($\tilde{\boldsymbol{G}}^{t^g}, \boldsymbol{E}^{\tilde{G}^{t^g}}$) function. The genetic information of the selected genes is altered through the application of genetic operations with the GeneticOperation($\boldsymbol{G}^{t^g+1}$) function. In this study, we used the four genetic operators proposed in [5], namely crossover, mutation, cut-and-paste, and break-and-join. The process is repeated until the values of t^i and t^g reach $T^i_{\max}$ and $T^g_{\max}$, respectively.

Upon completion, the optimal solution $\boldsymbol{G}^{best}$ is stored in $\hat{\boldsymbol{O}}$. The disassembly task labels $\hat{\boldsymbol{L}}$ are extracted from the datasets obtained from the input CAD model, using LabelExtraction($\hat{\boldsymbol{O}}$), at the commencement of the algorithm. The EEF contacts $\hat{\boldsymbol{C}}$, arm trajectories $\hat{\boldsymbol{T}}$, and object placement poses $\hat{\boldsymbol{P}}$ are extracted from the datasets generated through the robot simulations conducted at the beginning of the algorithm, utilizing OperationParameterExtraction($\hat{\boldsymbol{O}}$).

In the evaluation process, the feasibility and stability of each solution (sequence) $\boldsymbol{G}^{t^g}_i$ ($i = 1, \ldots, N^{gn}$) are determined using the aforementioned matrices, resulting in constraint satisfiability $\boldsymbol{C}^{G^{t^g}}$ consisting of binary elements that indicate whether each solution (sequence) is *feasible*, *stable*, and *available*. The number of genes is denoted by N^{gn}. A solution is considered available if it is both feasible and stable, *i.e.*, available := feasible $\wedge$ stable. The available solutions (sequences) are further sorted based on multiple criteria such as difficulty, efficiency, prioritization, and allocability. The evaluation provides evaluation values $\boldsymbol{E}^{G^{t^g}}$. The proposed MaOGA method resolves the minimization problem by employing multiple objective functions that are normalized between 0 and 1, with 0 representing the optimal value.

Table 1

Required labels for parts.

Label	Class
Task	screw, bolt, nut, plate, graspable, manual
Priority	value
Base	base
Ignore	ignore

3.2. CAD-informed matrices generations

This section outlines the preprocessing steps before the optimization loop, including the definition of part labels and the structure analysis of 3D CAD models. Matrix generation is also discussed, with a focus on determining constraints and calculating evaluation values. Part labels are assigned to each component of the model, as listed in Table 1. These labels include task labels, which link the parts to the EEFs of the robot; a priority label, which designates parts that should be disassembled preferentially; a base label, which is assigned to the root part fixed at the end of the sequence; and an ignore label, which is applied to parts that can be disregarded in the DSP as they are automatically disassembled during the removal of other parts.

The task labels are designated to specific parts, including screws, bolts, nuts, plates, graspable objects, and manually disassembled objects. The graspable label is assigned to parts other than screws, bolts, nuts, and plates, and can be manipulated by a two-finger gripper. The manual label is assigned to parts that are difficult to automate or necessitate special care when disassembling them. The priority and value labels are established for parts that merit prioritization during disassembly and are classified as valuable in terms of reuse, recycling, and remanufacturing.

To analyze the 3D CAD models (*e.g.*, label extraction), our framework utilizes pythonOCC,[3] which is a Python wrapper for the OpenCAS-CADE[4] library. The pythonOCC converts the standard for the exchange of product (STEP) model data into a format that allows for the extraction of part names, 3D poses, and shapes of the target product. The label for each part is obtained by splitting the part name using an underscore as the delimiter.

To obtain a quantifiable representation of part relationships, we employed previously proposed matrix representations [5,21]. Tariki et al. [5] utilized pythonOCC to generate the interference-free matrices X_j^{if} ($j = 1, \ldots, 6$) with elements indicating binary values of interference or interference-free between each set of two parts. Kiyokawa et al. [21] presented a method for generating a constraint degree matrix X^{cs} and contact matrix X^{ct}, which indicates the degree of constraint from 0 to 12 and the binary values of contact between each set of two parts. In this study, we develop constraint-free matrices X_j^{cf} ($j = 1, \ldots, 12$) and the motion feasibility matrices X_i^{mf} ($i = 1, \ldots, N^p$). As the total number of elements in the interference-free matrix, constraint degree matrix, contact matrix, constraint-free matrix, and motion feasibility matrix is $20 \times N^p \times N^p + \sum_{i=1}^{N^p} N_i^m \times N^p$, where N_i^m represents the number of feasible motions for each part P_i, it becomes increasingly imperative to develop efficient matrix calculation methods as the number of parts increases and the calculation time becomes exponentially more significant.

According to the definitions of an interference-free matrix, the matrix in each negative axis direction must be identical to the transposed matrix in each positive axis direction. The constraint degree matrix signifies the constraints existing between the different parts, and its order is immaterial. Hence, we can replicate the upper triangular components with its corresponding lower triangular components. Similarly, the lower triangular components of contact matrix and constraint

degree matrix can be derived from the corresponding upper triangular components.

To calculate the interference-free matrix and constraint degree matrix, we employ a simulation using a CAD model. Specifically, we place the two target parts in their assembled pose and check for interference when the target part is displaced in the axial direction relative to the object coordinate system. The target part is displaced by the length of the corresponding side of the bounding box that encompasses the two target parts unless interference is detected. Additionally, we conduct a displacement simulation to generate the constraint degree matrix by altering the pose of the target part according to the specified maximum clearance distance for the target product.

3.3. Constrained many-objective optimization

The figure depicted in Fig. 2 presents a proposed NSGA-III [15]-inspired MaOGA that displays a high level of performance in many-objective optimization problems with four or more objectives. The gene representation is initially arranged according to the user-defined number of genes (Step 1). To increase the number of effective initial solutions, the CCG is employed. The evaluation process for multiple objective functions is then carried out (Step 2). To incorporate the principles of NSGA-III, non-dominated sorting (Step 3) and reference-line-based gene selection (Step 4) are applied. The algorithm proceeds to generate the next chromosome (Step 6) and apply genetic operations (Step 7) unless the termination condition is met (Step 5).

The genetic encoding of information onto chromosomes and the subsequent genetic operations rely on existing methods, the efficacy of which has been demonstrated in the enhancement of ASP optimization [5]. These operations are subsequently applied, and the process returns to Step 2 and repeats until the specified termination condition is satisfied.

The four objective functions designed in this study are calculated from the information extracted from the 3D CAD model. This study considers a minimization problem that evaluates all objective functions equally, with 0 being the optimal evaluation value and 1 being the worst.

3.3.1. Constraints based on feasibility and stability

If the sequence is either infeasible or unstable, the disassembly operation cannot be executed by either human or robot. Conversely, if the sequence satisfies all constraints (available), then only those sequences that satisfy the constraints will be evaluated based on other objective functions.

The feasibility of the sequence is determined by checking both the order and motion feasibility. The order is considered order-feasible if it adheres to the specified conditions regarding the interference-free matrices X_j^{if} ($j = 1, \ldots, 6$):

$$\sum_{k=2}^{N^p} \left(\prod_{i=1}^{k-1} \sum_{j=1}^{6} X_j^{if}(P_{O_i}, P_{O_k}) > 0 \ ? \ 1 \ : \ 0 \right) = N^p - 1. \tag{1}$$

The order is considered motion-feasible if one or more collision-free and IK-solvable contacts and trajectories are found for all disassemblies needed to complete the sequence. Hence, the target sequence is determined as motion-feasible when the following conditions regarding the motion-feasibility matrices X_i^{mf} ($i = 1, \ldots, N^p$) are fulfilled.

$$\sum_{k=2}^{N^p} \left(\prod_{i=1}^{k-1} \sum_{j=1}^{N_{pok}^m} X_{pok}^{mf}(\mathcal{M}_j^{pok}, P_{O_i}) > 0 \ ? \ 1 \ : \ 0 \right) = N^p. \tag{2}$$

where $X_{pok}^{mf}(\mathcal{M}_j^{pok}, P_{O_i})$ represents the motion feasibility value between ith part P_{O_i} and the jth set of feasible motion $\mathcal{M}_j^{pok}$ for the target kth part P_{O_k} when pok represents P_{O_k}. A set of feasible motions $\mathcal{M}_j^i$ comprises feasible contact C_j^i, feasible trajectories $\mathcal{T}_j^i$, and feasible object placement $\mathcal{P}_j^i$. The number of feasible motions generated for each part is represented by N_i^m ($i = 1, \ldots, N^p$).

The evaluation of stability involves the assessment of two distinct criteria. The first criterion pertains to whether the parts remaining after removal of the target part can maintain an upright posture in the workplace (upright condition). The second criterion involves determining whether all parts are interconnected (connection condition). The upright condition (static stability) can be easily assessed using a method established in the fields of optimal 3D fabrication [50] and balance control of humanoid robots [51]. If flexible fixtures are available to hold various poses of the parts, the upright condition can be disregarded, as it will always be satisfied. In this study, we employ an array of multiple soft-jigs [52] as a solution to address this issue. The connection condition can be easily examined by analyzing the constituent elements. The sequence is considered stable when the following conditions are fulfilled:

$$\sum_{k=2}^{N^p}\left(\sum_{i=1}^{k-1} X^{ct}(P_{O_i}, P_{O_k}) \neq 0 \; ? \; 1 \; : \; 0\right) = N^p - 1. \tag{3}$$

The symbol $\boldsymbol{X}^{ct}$ denotes a contact matrix, where a value of one indicates the presence of a nonzero value in the constraint degree matrix $\boldsymbol{X}^{cs}$.

3.3.2. Initialization of chromosomes

To increase the number of high-quality initial solutions generated, we employ stability-based chromosome initialization utilizing the CCG depicted in Fig. 3(a). The graph is automatically generated using the following procedure:

(1) The parts are classified as either bolts or screws (fixing parts, represented by box-shaped nodes) or other parts (non-fixing parts, represented by circle-shaped nodes) based on the task labels extracted through model structure analysis.
(2) Edges are generated between each node by analyzing the contact matrix, connecting nodes that are in contact with each other.
(3) The edges connecting to the fixing part nodes are categorized and assigned as connection edges (red-colored edges) and other edges (black-colored edges).

The numbers in the nodes correspond to the part index of the disassembly target product.

The following procedure is performed based on the generated CCGs:

(1) The base-labeled part or the largest part is designated as the root node.
(2) The distance (minimum number of edges) from the root node to each node is calculated.
(3) A node is randomly selected from the set of nodes at the maximum distance.
(4) If the selected part is a fixing part, it is placed at the beginning of the sequence. If it is a non-fixing part, a neighboring part connecting the selected part to another part is randomly selected and placed at the beginning of the sequence. If there are no fixing parts, the selected non-fixing part is placed at the beginning of the sequence.
(5) Steps 2 to 4 are repeated until only the root node remains. The part displaying the root node is then placed at the end of the disassembly sequence, resulting in the end of the generation process.

Fig. 3(b) shows snapshots of an example of the disassembly procedure. The absence of isolated nodes not connected to any edge indicates that every disassembly can be regarded as stable.

The essential prerequisite for attaining an interference-free sequence when dealing with a fully constrained part restricted from moving in all 12 directions is the prioritized removal of the fixing part. Therefore, utilizing CCGI is more advantageous for generating interference-free initial solutions than relying on random initialization.

In other words, the use of CCGI is more likely to result in the generation of an available sequence.

Subsequently, the optimization process commences with the initial solutions. Throughout the optimization procedure, feasible and stable solutions also improve the evaluation of the objective functions. To ensure uniformly evaluated multiple objectives, the objective values must be normalized, enabling meaningful distance metric computations in the objective space. Experimental results from a previous study by Blank et al. [53] indicated that normalization affects the performance of evolutionary multi-objective optimization algorithms. Thus, this study normalizes the four objectives to values ranging from zero to one, as described in the following sections.

3.3.3. Difficulty

Among the various difficulty definitions [54], we propose a specific definition for the objective function that corresponds to the constraint state transition difficulty [55], which is a type of order difficulty. This can be expressed as follows:

$$f_d := \begin{cases} H/12(N^p - 1) & \text{if } \boldsymbol{O} \text{ is available} \\ 1 & \text{otherwise} \end{cases}. \tag{4}$$

where H denotes the maximum level of constraint state transition difficulty associated with the disassembly of each individual part.

$$H := \max_{k \in \{2,3,\dots,N^p\}} \sum_{i=1}^{k-1} X^{cs}(P_{O_i}, P_{O_k}) \; < 12(N^p - 1). \tag{5}$$

$\sum_{i=1}^{k-1} X^{cs}(P_{O_i}, P_{O_k})$ represents the total constraint degree imposed on the kth part P_{O_k} and its undisassembled parts $P_{O_1}, P_{O_2}, \dots, P_{O_{k-1}}$. In accordance with the established definition, the maximum constraint degree between the two parts is 12, consequently, each element of the constraint degree matrix $\boldsymbol{X}^{cs}$ is calculated as

$$X^{cs}(P_i, P_k) = 12 - \sum_{j=1}^{12} X_j^{cf}(P_i, P_k) \; \in \{0, \dots, 11\}. \tag{6}$$

The matrix $\boldsymbol{X}_j^{cf}$ $(j = 1, \dots, 12)$ represents the constraint-free matrix.

3.3.4. Efficiency

The task labels are utilized to maximize the efficiency of the sequence of tasks by minimizing the number of task changes and the distance between the center of mass of the target parts.

$$f_e := \begin{cases} [N^{tc}/(N^p - 1) \\ \quad + D/(N^p \times D_{\max})]/2 & \text{if } \boldsymbol{O} \text{ is available} \\ 1 & \text{otherwise} \end{cases}. \tag{7}$$

The number of task changes N^{tc} can be determined by analyzing the task labels T_{O_i} of each part $i = 1, \dots, N^p$.

$$N^{tc} = \sum_{k=2}^{N^p} \left[(T_{O_k} = T_{O_{k-1}}) \; ? \; 1 \; : \; 0 \right]. \tag{8}$$

The total moving distance D can be determined by measuring the distance d_{P_i, P_j} between each part.

$$D = \sum_{k=2}^{N^p} d_{P_{O_k}, P_{O_{k-1}}}. \tag{9}$$

The maximum distance between any two parts is denoted by $D_{\max}$.

3.3.5. Prioritization

The objective function for prioritization is defined as follows:

$$f_p := \begin{cases} 1 - R/\prod_{l=N^p-N^{pp}}^{N^p} l & \text{if } \boldsymbol{O} \text{ is available} \\ 1 & \text{otherwise} \end{cases}. \tag{10}$$

N^{pp} denotes the number of prioritized parts. The degree of prioritization R is determined by the positions of priority parts.

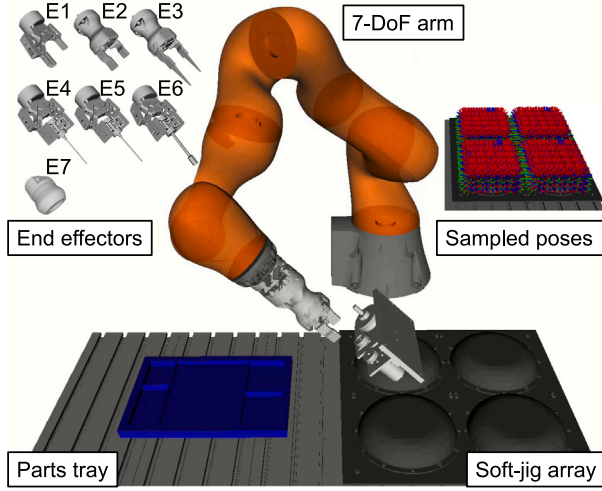$$R = \sum_{m=1}^{N^{pp}} O_m^p. \tag{11}$$

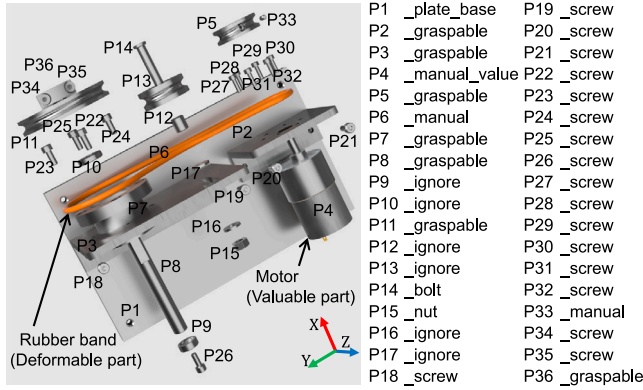**Fig. 4.** Robotic disassembly setup for simulation experiments.

P1	_plate_base	P19	_screw
P2	_graspable	P20	_screw
P3	_graspable	P21	_screw
P4	_manual_value	P22	_screw
P5	_graspable	P23	_screw
P6	_manual	P24	_screw
P7	_graspable	P25	_screw
P8	_graspable	P26	_screw
P9	_ignore	P27	_screw
P10	_ignore	P28	_screw
P11	_graspable	P29	_screw
P12	_ignore	P30	_screw
P13	_ignore	P31	_screw
P14	_bolt	P32	_screw
P15	_nut	P33	_manual
P16	_ignore	P34	_screw
P17	_ignore	P35	_screw
P18	_screw	P36	_graspable

Fig. 5. Model appearance and assigned parts labels.

O_m^p represents the ordinal position of the mth priority part.

3.3.6. Allocability

Allocability is based on the sequential position of the manually labeled parts to be disassembled.

$$f_a := \begin{cases} |O_r^m - O_l^m|/(N^p - 1) & \text{if } \boldsymbol{O} \text{ is available} \\ 1 & \text{otherwise} \end{cases}. \tag{12}$$

O_l^m and O_r^m indicate the latest and earliest ordinal positions of manually disassembled parts, respectively.

4. Experiments of robotic DSP

4.1. Overview

Our experiments verified the efficacy of the proposed method in terms of structure analysis, matrix generation, and DSP using the robotic disassembly setup illustrated in Fig. 4. The objective of our experiments was to evaluate the performance of the proposed method on a belt drive unit used in an assembly challenge [56]. Fig. 5 depicts the appearance of the CAD model and shows the labels assigned for our experiments.

The product disassembly system incorporated a seven-degree-of-freedom (DoF) arm and various EEFs. Three different types of two-finger parallel grippers were utilized: the Robotiq 2F-85 (E1), the Robotiq HandE (E2), and the Robotiq HandE with longer fingers (E3).

In order to enable the robot arm to screw bolts and screws using a two-finger parallel gripper, Hu et al. [57] developed a mechanical screwing tool. The system utilized three configurations of the screwing tool with different tool tip parts: M3 hex wrench (E4), M4 hex wrench (E5), and M6 socket wrench (E6). The suction gripper employed was the CONVUM SGB30 (E7), also known as the balloon hand. It is well suited for a wide range of workpiece shapes and sizes, capable of handling even uneven or heavy workpieces.

The process of structural analysis involves identifying the labels assigned to various parts. The belt drive unit comprises the labels of screw, bolt, nut, plate, graspable, manual, value, base, and ignore. The P4 motor was regarded as a valuable part, hence it possesses a value label in addition to its manual label, owing to its delicate disassembly process. The rubber belt P6 and hexagon socket set screw P35 were also assigned a manual label because of their difficulty in robotic disassembly. The spacers P9, P10, and P12, pulley P13, and washers P16 and P17 were assigned the ignore label, as they will naturally come off during the disassembly process of other parts.

The allocation of the seven types of EEFs was determined according to the task label and shape features for 27 of the 36 parts. These 27 parts, namely P1, P2, P3, P5, P7, P8, P11, P14, P15, P18, P19, P20, P21, P22, P23, P24, P25, P26, P27, P28, P29, P30, P31, P32, P34, P35, and P36 did not have a manual or ignore label. Therefore, for these parts, we have assigned E7, E2, E2, E2, E2, E1, E2, E5, E6, E5, E5, E5, E5, E5, E5, E5, E4, E4, E4, E4, E4, E4, E5, E5, and E3.

The generation of sequences requires several parameters $\boldsymbol{P}^{ga}$, including the number of chromosomes, crossover rate, mutation rate, cut-and-paste rate, and break-and-join rate, which are identical to those utilized in [21]. In our methodology, we set the number of generation updates to 500, and the number of iterations to 10. To create robot motions, we utilize a contact planning software[5] that uses an object-geometry-based approach to potential contact generation, as described in [58]. The robot arm trajectory planner relies on the RRT-connect algorithm [59], which was implemented in MoveIt! motion planning framework[6] of robot operating system (ROS), as well as IKFast [60] for solving the kinematics. In a work environment equipped with four soft-jigs arranged on the workspace, as depicted in the upper right corner of Fig. 4, during the processing of RobotSimulation($\boldsymbol{M}$) in Algorithm 1, the target assembled parts were positioned and oriented in the soft-jig array. Thereafter, a trajectory was explored to identify potential collision-free contact points and efficiently executable trajectories for the target product at these specific positions and orientations.

4.2. Results of CAD-informed matrices generations

In order to generate multiple matrices for determining the constraints and calculating the evaluation values in the optimization, we initially conducted a thorough analysis of the 3D CAD models. Our analysis successfully extracted all part labels with a high degree of accuracy. The center of mass and pose parameters obtained from the STEP model were accurately extracted with 100% accuracy, as illustrated in Fig. 6(a). The interference check between parts for matrix generation is shown in Fig. 6(b), where the red highlighted area indicates the interfered volume between the base plate part P1 and L-shaped plate part P2.

We assessed the performance of the automatic matrix extraction based on accuracies. The constraint degree matrix contains positive integers, and the calculation was regarded as successful when a positive integer was correctly determined to match the manually annotated value.

The generation accuracies of the interference-free and contact matrices were 98.9% and 96.8%, respectively. The success rates of the
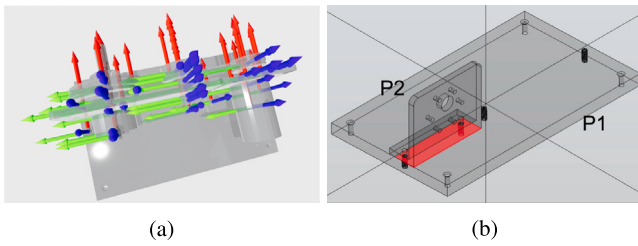

Fig. 6. Results of structure analysis and interference check for the belt drive unit. (a) Center of mass and pose. (b) Example of interference.
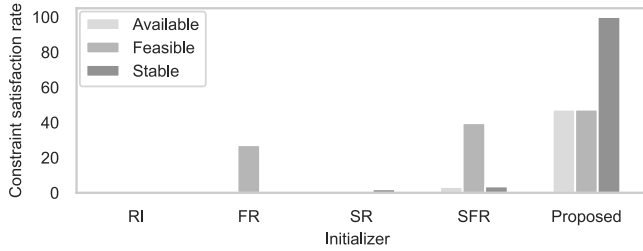

Fig. 7. Success rates of finding constraint-satisfied sequences in 1000-trial initialization [%]. RI, FR, SR, and SFR are comparative methods that show random initialization, feasibility-based rearrangement, stability-based rearrangement, and stability-and-feasibility-based rearrangement methods, respectively.

constraint degree and constraint-free matrices were 90.2% and 92.9%, respectively, which were not low. The interference checks based on the displacement simulation can sometimes fail due to the limitations of the Boolean operation performance. The Boolean operations between curved surfaces can be challenging and may result in errors. In the future, it may be necessary to consider a method for directly estimating the degree of constraint based on the shape.

4.3. Optimizing sequences

4.3.1. Performance of Chromosome initialization

We undertook a comparative analysis of chromosome initialization methods. To this end, we devised three methods for initializing chromosomes: random initialization (RI), repeated sequence changes to minimize interference (FR) [5], and repeated sequence changes to maximize stability (SFR). Fig. 7 shows the result comparisons. The bars depict the mean values of the feasible, stable, and available (feasible and stable) rates [%] for the 1000-trial initialization. Although the rates of feasible solutions with FR and SFR were 27.1% and 39.6%, respectively, the rates of stable solutions were 0.2% and 3.6% for FR and SFR, which resulted in available rates of 0.1% and 3.3%, respectively.

Nevertheless, the feasibility, stability, and availability rates for the proposed method were 47.3%, 100%, and 47.3%, respectively. The results indicate that the proposed method can generate a feasible and stable sequence when compared to other methods.

4.3.2. Performance of optimization

The generated order $\hat{O}$ composed of elements 1, 3, 19, 18, 14, 15, 7, 8, 24, 22, 23, 25, 11, 36, 34, 35, 26, 2, 20, 21, 4, 29, 28, 27, 31, 32, 30, 5, 33, 6 attained the lowest evaluation value among all evaluated sequences. This sequence includes parts with manual labels, but excludes those with the ignore label. This sequence is a feasible and stable (available) solution that adheres to all imposed constraints.

Fig. 8 shows the generated sequence. As can be observed in the figure, the unconstrained parts P6 and P33 with manual labels, are situated at the commencement of the sequence. In addition, motor

part P4 with manual and value labels, is disassembled at the earliest possible timing following the elimination of all constraining parts, namely P5, P30, P32, P31, P27, P28, and P29. It is worth noting that other arrangements also fulfill order feasibility, motion feasibility, and stability, while simultaneously exhibiting low difficulty and high efficiency, and adhering to label-defined prioritization.

Fig. 9(a) presents the performance comparison results. We conducted ten iterations of 500 generation updates for the optimization loop. Fig. 9(a) shows the mean values of the available, feasible, and stable rates. Fig. 9(b) shows the mean and standard deviation of the evaluation values for each objective function. The bars represent the mean values and error bars indicate the standard deviations. The 'w/o CCGI' method does not utilize CCGI but instead uses an initialization method that considers only feasibility, as previously described in [5]. The 'w/o NSGA-III' result is based on the NSGA-II-inspired algorithm proposed in [21]. The 'w/o f_d', 'w/o f_e', 'w/o f_p', and 'w/o f_a' denote the optimization results excluding each of the objective functions of Eqs. (4), (7), (10), and (12).

The effectiveness of the proposed initialization method in consistently producing available solutions was demonstrated by the lack of solution generation when the CCGI method was not employed, resulting in a 0% rate of available solution generation. Comparing the proposed method with the 'w/o NSGA-III' method, both achieved 100% success rates in generating available solutions. When evaluating the performance of each method based on the four objective functions, Difficulty, Efficiency, Prioritization, and Allocability, the mean evaluation values of both of them are almost the same.

The visualizations in Fig. 10 display the mean evaluation values of the final solutions after each optimization iteration. For the petal chart, smaller petals indicate better evaluation values. In the case of the radar chart, a smaller square area signifies a better evaluation value. The graphs (original) show the values calculated using f_d, f_e, f_p, and f_a. The graphs for the relative evaluation values f_d^*, f_e^*, f_p^*, and f_a^* were min–max normalized to scale the maximum value of each objective function in all methods except 'w/o CCGI' to 1.0. Fig. 11 depicts the transitions of constraint satisfaction rates for the 'w/o NSGA-III' and proposed methods. Fig. 12 depicts the learning curves for the four objective functions evaluated in the cases of using their two methods.

As depicted in the charts presented in Fig. 10(a) and (c), the performance of the proposed method and other comparative methods may not exhibit significant disparities. In fact, the sum of our evaluation values (original) for 'w/o CCGI', 'w/o Difficulty', 'w/o Efficiency', 'w/o Prioritization', 'w/o Availability', and Proposed amounted to 4.0, 0.918, 0.934, 1.22, 1.07, and 1.01, respectively. On the other hand, the standard deviations of the normalized evaluation values were 0.073, 0.180, 0.253, 0.158, 0.129, and 0.042, respectively. Notably, the proposed method exhibited the lowest standard deviation. This result indicates that the proposed method was effective in optimizing the system while simultaneously evaluating four objective functions.

As evidenced in Fig. 11, although both methods achieved a constraint satisfaction rate of 100% by the 5-th generation, the proposed method demonstrated a more rapid improvement in these rates up to that point. Additionally, as depicted in Fig. 12, the proposed method achieves a smoother and more consistent learning curve compared to the 'w/o NSGA-III' method, suggesting that the proposed method consistently and stably reduced the evaluation values of the four objective functions throughout the learning process. These findings suggest that the NSGA-III-inspired algorithm is beneficial for facilitating convergence in the learning process. In forthcoming studies, we intend to further explore the effectiveness of this feature by applying the proposed method to a variety of other target objects.

Fig. 13 shows a comparison of performance using methods that employ a single objective function. The bars above the titles of 'w/ Difficulty', 'w/ Efficiency', 'w/ Prioritization', and 'w/ Allocability' represent the mean values of the four objective functions when optimizing the solutions under constraints and a single objective function of f_d,
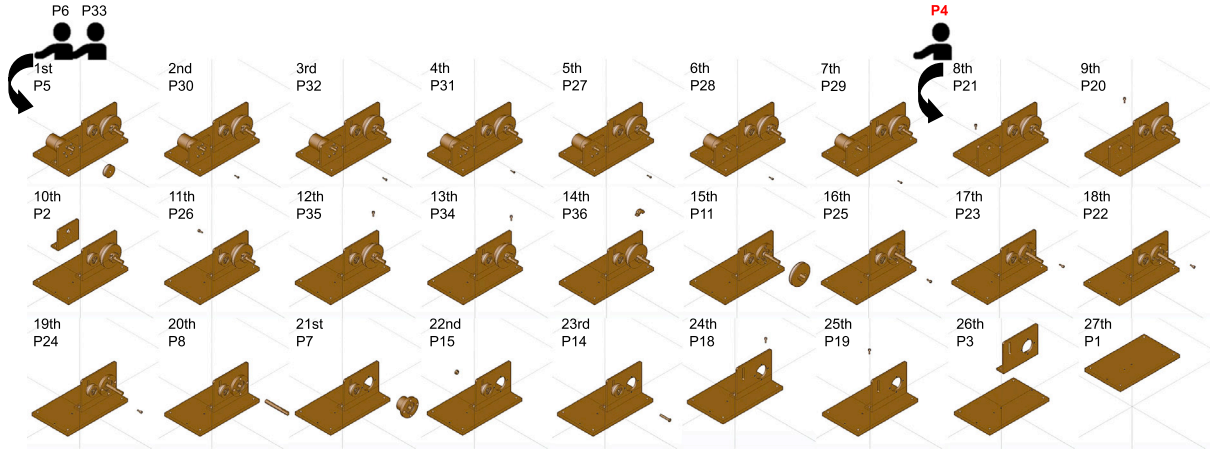
Fig. 8. A sequence determined after the optimization. The disassemblies shown in the snapshots are order-feasible (interference-free) and stable.
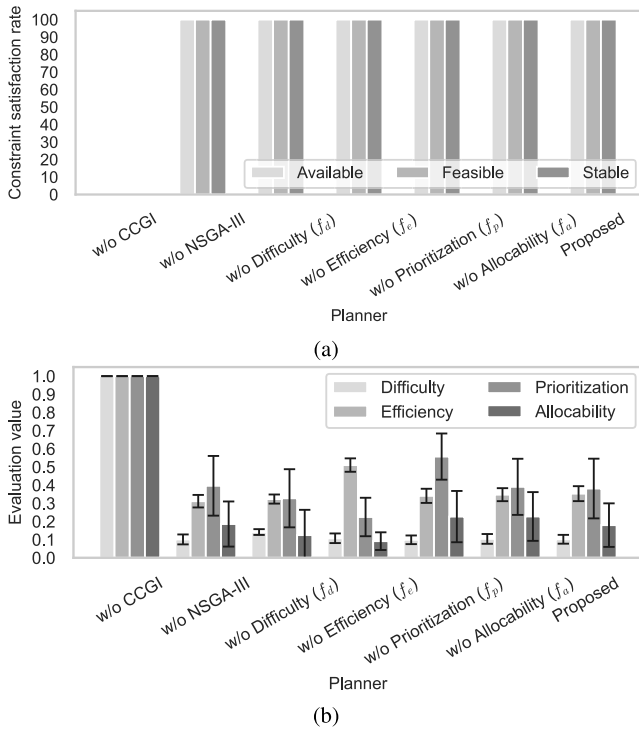

Fig. 9. The many-objective optimization results of 500-generation and 10-iteration optimization. (a) Percentages of constraint-satisfied solutions [%]. (b) Mean ± standard deviation of the evaluation values.

f_e, f_p, and f_a, respectively. The error bars indicate standard deviations. The evaluation value using f_d for 'w/ Difficulty' method is 0.075, which is the lowest values compared to those of other methods. The difficulty evaluation values of 'w/ Efficiency', 'w/ Prioritization', and 'w/ Allocability' methods are 0.112, 0.147, and 0.142, respectively. As shown in Fig. 13, the other three evaluation values exhibited the same trend. The results shown in Fig. 13 demonstrate that the proposed algorithm can effectively perform single-objective optimization, allowing users to choose the objective function that they wish to prioritize.

4.4. Feasibility of robotic disassembly

Fig. 14 illustrates the feasible contacts by the task-tailored EEFs. The EEFs colored in green indicate successful contacts, while those colored in red represent failed contacts resulting from collisions. We chose the

approachable contact from the robot's EEF pose among the feasible contacts $\mathcal{C}$.

Fig. 15 illustrates the generated feasible (collision-free and IK-solvable) motions for the optimal sequence for robotic disassembly. Fig. 15 includes snapshots of the generated EEF contact $\hat{\mathcal{C}}$, arm trajectory $\hat{\mathcal{T}}$, and object placement pose $\hat{\mathcal{P}}$ fixed on the soft-jig array. For each disassembly, the three pictures show the arm in a pose at the post-contact position, the zoom of the post-contact pose, and the placement pose. It is crucial to place the robot in such a position that the target part falls within the arm's movable range. In some cases, moving the target part to the arm side using a turntable or a flexible fixture placed beneath the object can be effective even for large objects. Our solution was to use a soft-jig, which was previously developed in [52,61,62]. The specific approach may vary based on the configuration of the robot arm, target part, and workspace, utilizing a soft-jig array such as ours (Fig. 4), which can fix the target parts in various positions and orientations, enabling the generation of a greater number of trajectory candidates that facilitate the arm approach to the target part.

Although this study did not specifically focus on robot motion planning for real-world tasks, the findings nonetheless revealed the capability to devise feasible contact for all parts using their corresponding EEFs, as well as the corresponding trajectory for all pick-and-place tasks, spanning the entire sequence.

5. Discussions

5.1. Efficient and robust matrix generation

The time required for each module in this study was 12.7 s, 16400 s (4.57 h), and 1050 s (17.5 min) for model structure analysis, matrix generation, and sequence exploring, respectively. The computational time depends on the size of the STEP file (13.8 [MB]) and the number of parts (36 parts). Specifically, when multiple fine curved surfaces, such as fillets, are represented precisely, the cost of calculation becomes excessive when conducting Boolean operations to assess interferences among them.

The reliability of matrix generation depends heavily on the robustness of these Boolean operations. In future work, to mitigate the computational burden and potential failures, we aim to develop a method that simplifies the shape and reduces computational costs. Utilizing the latest semantic shape representations [63,64] or coarse collision proxies for initial pruning can drastically reduce the frequency of exact Boolean checks.

Similarly, the lack of an automatic labeling pipeline limits out-of-the-box applicability. Currently, we utilize manual input for explicit labels to ensure accuracy. However, to minimize the burden on the model designer, it is desirable to develop a method for automatically
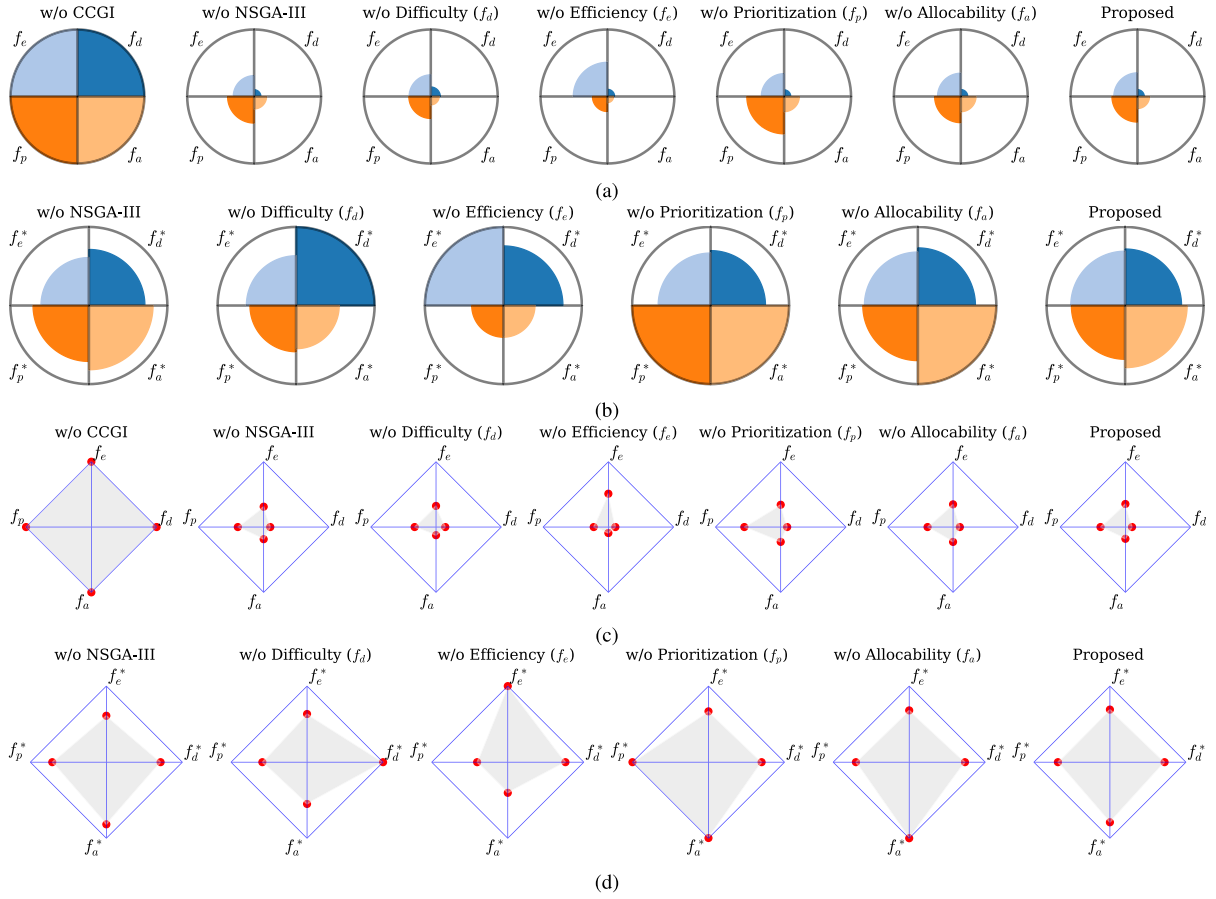
Fig. 10. Visualization of the mean evaluation values over the generation updates for all iterations. (a) Petal chart (original). (b) Petal chart (relative evaluation). (c) Radar chart (original). (d) Radar chart (relative evaluation). For the petal chart, the smaller each petal, the better is the evaluation value. Regarding the radar chart, the smaller the square area, the better the evaluation value. The graphs for the relative evaluation were min–max normalized so that the maximum value of each objective function in all methods except 'w/o CCGI' was scaled to 1.0.

extracting labels from geometric information. The PointNet series [65–67] has demonstrated the potential to classify 3D shape data. Integrating such learned classifiers to directly infer task and base labels from geometric characteristics represents a viable solution to enhance the framework's adaptability.

5.2. Optimization algorithm

To further improve convergence speed and optimize solution quality, the static parameters used in the current MaOGA (*e.g.*, crossover and mutation rates) could be dynamically adjusted. The incorporation of adaptive parameter tuning, which regulates exploration and exploitation balance based on the generation progress, is a promising strategy to accelerate the search process.

Additionally, computational efficiency can be enhanced by focusing on user-preferred regions of the Pareto front. Several extended NSGA-III algorithms have been proposed, such as U-NSGA-III [68], which incorporates tournament pressure to achieve unified and improved performance. Furthermore, R-NSGA-III [69] extends the procedure by introducing reference points according to user-supplied aspiration points.

Our experimental results indicate that there may be varying levels of heterogeneity in the balance of each objective function depending on the target product. Therefore, it may be beneficial to first use a general NSGA-III search to identify these heterogeneities and then use a combination of U-NSGA-III and R-NSGA-III to efficiently optimize them in a narrower search space. This focused approach allows the algorithm to converge more rapidly to high-quality solutions, avoiding the computational cost of approximating the entire Pareto-optimal front.

5.3. Scalability

The matrix generation process currently imposes a computational burden. However, in the context of offline process planning for re-manufacturing, this duration serves as a one-time setup cost [24]. This characteristic mitigates the impact of the initial calculation time, clarifying its validity for high-volume scenarios where the optimized sequence is repeatedly executed for identical products.

To further ensure scalability for larger assemblies or more complex products, we can employ strategies beyond the shape simplification. First, the interference checks, which dominate the computation, can be accelerated through parallel computing since they are computationally independent for each part pair. Second, adopting a hierarchical approach by decomposing a large assembly into subassemblies can effectively manage the combinatorial complexity of the sequence optimization [27]. Additionally, integrating learning-based methods to infer removability or approximate collision checks [29,30] offers a promising direction to significantly reduce the frequency of exact Boolean operations required during the initial planning phase.

5.4. Real-world disassembly

The proposed approach relies on geometric reasoning within a simulated environment, assuming rigid bodies and perfect perception. However, shifting to real-world scenarios introduces physical uncertainties, such as sensor noise, friction, and part deformation. Therefore, the validity of the generated sequence lies in its role as a high-level nominal plan. To bridge the gap between simulation and reality (Sim2Real),
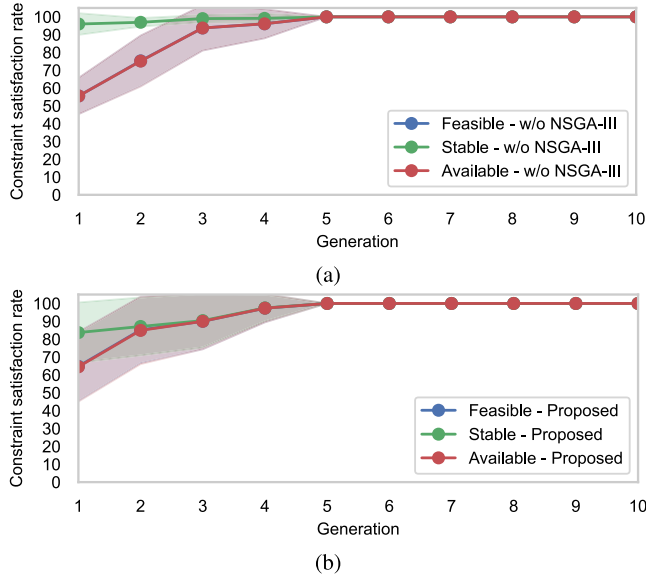
Fig. 11. Changes in the mean values (over 10 iterations) of the constraint satisfaction rates [%] during 500 generation updates in the first iteration. (a) 'w/o NSGA-III'. (b) Proposed. After the fifth generation, all constraint satisfaction rates were 100%. The translucent bands around the lines illustrate the standard deviations (over 10 iterations).
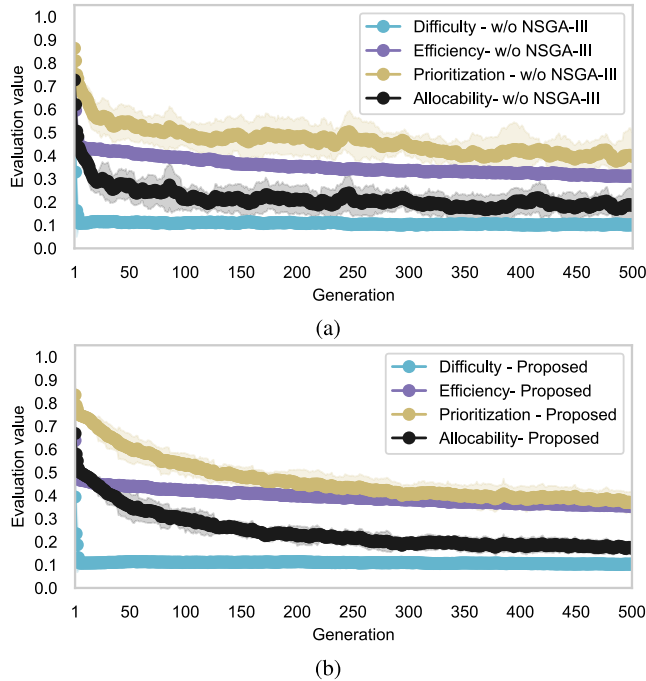

Fig. 12. Changes in the mean evaluation values (over 10 iterations) during 500 generation updates in the first iteration. (a) 'w/o NSGA-III'. (b) Proposed. The translucent bands around the lines illustrate the standard deviations (over 10 iterations).

the geometric plan must be coupled with robust low-level execution strategies.

In contrast to the process of assembly, which requires precise and meticulous operations, disassembly can sometimes tolerate minor damage to target products, with the exception of high-value parts that
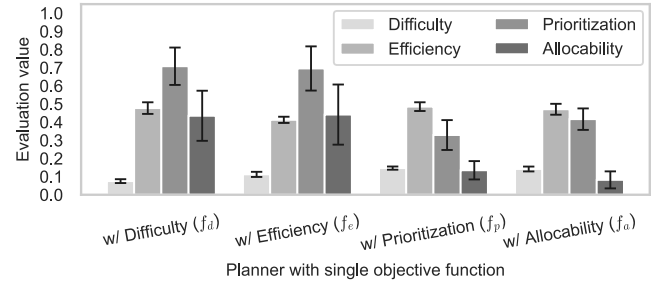

Fig. 13. Final evaluation values of single-objective optimization. The mean ± standard deviation of the evaluation values ranged from 0 to 1 in the 500-generation and 10-iteration optimizations [%].

must be carefully disassembled. The generation of robotic disassembly operations in simulations, such as cutting or crushing flexible objects or removing parts without disassembling fasteners, presents a significant challenge. A promising new approach to self-supervised learning in a real-world environment holds potential for acquiring disassembly tasks that involve breaking and damage. Future studies will consider constructing such an approach.

On the other hand, there are situations in which every part must be carefully disassembled without compromising the quality of the materials or breaking them. The precise disassembly operations can be achieved by following the reverse sequence of the assembly operations. In recent years, reinforcement learning approaches have garnered attention for enabling robots to perform contact-rich manipulation tasks in real-world environments, thereby bridging the gap between simulation and reality [70]. It may be possible to address the learning problem of the disassembly action policy inference model in a similar manner, by following the analogy between assembly and disassembly.

5.5. HRC-oriented DSP

The proposed method for DSP is not limited to the domain of robot motion generation but may also prove effective in teaching sequences to human workers. By selecting the desired product model using a tablet computer, the results of the automatic DSP can be presented as a guide to efficiently determine the order of the disassembly parts.

This study represents an initial effort to simplify sequencing the semi-automated disassembly operations. While other criteria for evaluating sequences exist, such as those organized in [71], this study focused solely on four perspectives: difficulty, efficiency, prioritization, and allocability. As not all objective functions created based on the provider's motivation can be validated, this study sought a solution for one example design using an evolutionary many-objective optimization algorithm. Kiyokawa et al. [54] provide further definitions of the difficulty and complexity.

Previous studies have explored the application of graph representation to determine the necessary operations, tasks, motions, arms, and tools for cooking and furniture assembly sequences using robots [72–75]. The use of graph-based methods for determining arm and tool availability, calculating efficiency, and determining difficulty levels at different stages represents a promising direction for developing a more general disassembly sequence planner. If we could design the method to encode the graph into the genes and genetic operations based on the encoded representation, it could potentially be accomplished.
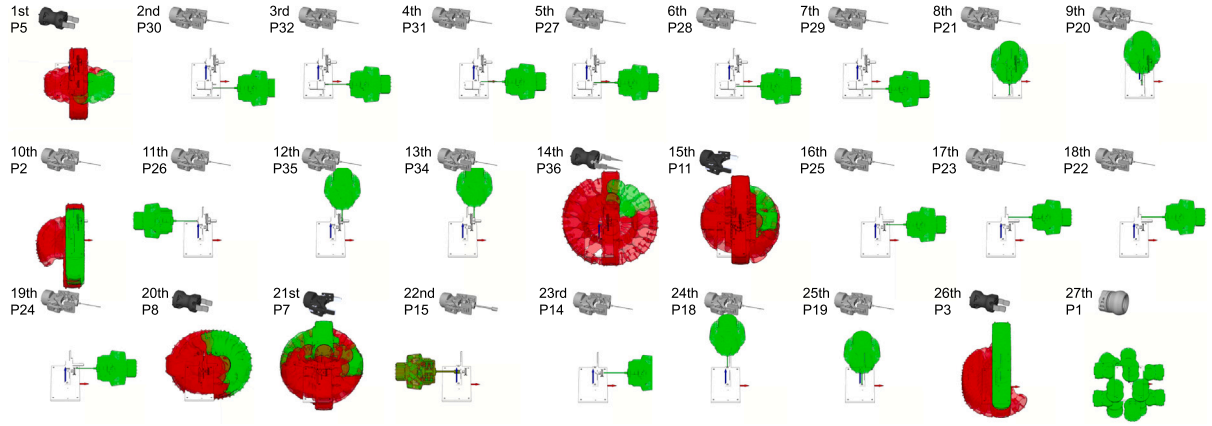
Fig. 14. The generated feasible contacts C for each target part by the EEFs. The green EEFs show collision-free contacts that do not interfere with the target object. The red circles show the possible failed contacts owing to collisions. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Fig. 15. The generated feasible (collision-free and IK-solvable) motions include $\hat{C}$, $\hat{T}$, and $\hat{P}$. The three pictures for each disassembly show the arm in a pose at the post-contact position, the zoom of the post-contact pose, and the placement pose.

6. Conclusion

This study focused on disassembly sequence planning (DSP) that incorporates semi-automated robotic operations. The proposed robotic DSP method uses an evolutionary many-objective optimization algorithm, namely NSGA-III-inspired MaOGA that iteratively updates generations and evaluates them with multiple objective functions and constraints.

The results of the disassembly sequence planning for a mechanical product with 36 parts showed that the proposed method can find a Pareto optimal solution oriented towards semi-automated robotic operations. The algorithm successfully generated a sequence that satisfied the feasibility, stability, and improved conditions in terms of difficulty, efficiency, prioritization, and allocability functions.

Specifically, the use of contact and connection graph (CCG)-based initialization allows for the repeatable generation of a large number of available initial solutions, whereas the algorithm utilized nondominated sorting and niching with reference lines to encourage steady and stable exploration of the solutions and uniformly lower overall evaluation values. The final solution featured interference-free, stable, efficient, easy-to-handle, correctly prioritized, and non-redundant task-assigned order that enables robots collision-free, IK-solvable, and efficient motions in the context of semi-automated robotic disassembly operations.

CRediT authorship contribution statement

Takuya Kiyokawa: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Kensuke Harada:** Writing – review & editing, Supervision, Project administration, Funding acquisition. **Weiwei Wan:** Visualization, Software. **Tomoki Ishikura:** Resources, Investigation, Data curation. **Naoya Miyaji:** Resources, Project administration, Funding acquisition, Data curation. **Genichiro Matsuda:** Supervision, Resources, Project administration, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Takuya Kiyokawa reports financial support was provided by New Energy and Industrial Technology Development Organization. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.robot.2025.105301.

References

[1] H. Poschmann, H. Brüggemann, D. Goldmann, Disassembly 4.0: A review on using robotics in disassembly tasks as a way of automation, Chem. Ing. Tech. 92 (4) (2020) 341–359.

[2] M. Daneshmand, F. Noroozi, C. Corneanu, F. Mafakheri, P. Fiorini, Industry 4.0 and prospects of circular economy: a survey of robotic assembly and disassembly, Int. J. Adv. Manuf. Tech. 124 (2022) 1–28.

[3] S. Lou, R. Tan, Y. Zhang, C. Lv, Human-robot interactive disassembly planning in industry 5.0*, in: Proc. IEEE/ASME Int. Conf. Adv. Intell. Mech., 2023, pp. 891–895.

[4] G. Thomas, M. Chien, A. Tamar, J.A. Ojea, P. Abbeel, Learning robotic assembly from CAD, in: Proc. IEEE Int. Conf. Robot. Autom., 2018, pp. 3524–3531.

[5] K. Tariki, T. Kiyokawa, T. Nagatani, J. Takamatsu, T. Ogasawara, Generating complex assembly sequences from 3D CAD models considering insertion relations, Adv. Robot. 35 (6) (2021) 337–348.

[6] F. Chervinskii, S. Zobov, A. Rybnikov, D. Petrov, K. Vendidandi, Auto-Assembly: a framework for automated robotic assembly directly from CAD, in: Proc. IEEE Int. Conf. Robot. Autom., 2023, pp. 11294–11300.

[7] Y. Koga, H. Kerrick, S. Chitta, On CAD informed adaptive robotic assembly, in: Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2022, pp. 10207–10214.

[8] M. Goldwasser, J.-C. Latombe, R. Motwani, Complexity measures for assembly sequences, in: Proc. IEEE Int. Conf. Robot. Autom., 2, 1996, pp. 1851–1857.

[9] W.H. Chen, K. Wegener, F. Dietrich, A robot assistant for unscrewing in hybrid human-robot disassembly, in: Proc. IEEE Int. Conf. Robot. Biomim., 2014, pp. 536–541.

[10] S. Parsa, M. Saadat, Human-robot collaboration disassembly planning for end-of-life product disassembly process, Robot. Comput.-Integr. Manuf. 71 (2021) 102170.

[11] H.-y. Liao, Y. Chen, B. Hu, S. Behdad, Optimization-based disassembly sequence planning under uncertainty for human–robot collaboration, J. Mech. Des. 145 (2) (2022) 022001.

[12] S. Hjorth, E. Lamon, D. Chrysostomou, A. Ajoudani, Design of an energy-aware cartesian impedance controller for collaborative disassembly, in: Proc. IEEE Int. Conf. Robot. Autom., 2023, pp. 7483–7489.

[13] X. Guo, C. Fan, M. Zhou, S. Liu, J. Wang, S. Qin, Y. Tang, Human–robot collaborative disassembly line balancing problem with stochastic operation time and a solution via multi-objective shuffled frog leaping algorithm, IEEE Trans. Autom. Sci. Eng. (2023) 1–12.

[14] M.-L. Lee, W. Liu, S. Behdad, X. Liang, M. Zheng, Robot-assisted disassembly sequence planning with real-time human motion prediction, IEEE Trans. Syst. Man Cybern. Syst. 53 (1) (2023) 438–450.

[15] K. Deb, H. Jain, An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints, IEEE Trans. Evol. Comput. 18 (4) (2014) 577–601.

[16] L. Homem de Mello, A. Sanderson, Planning repair sequences using the AND/OR graph representation of assembly plans, in: Proc. IEEE Int. Conf. Robot. Autom., 1988, pp. 1861–1862.

[17] S. Lee, H. Moradi, Disassembly sequencing and assembly sequence verification using force flow networks, in: Proc. IEEE Int. Conf. Robot. Autom., vol. 4, 1999, pp. 2762–2767.

[18] S. Sundaram, I. Remmler, N. Amato, Disassembly sequencing using a motion planning approach, in: Proc. IEEE Int. Conf. Robot. Autom., vol. 2, 2001, pp. 1475–1480.

[19] D.T. Le, J. Cortes, T. Simeon, A path planning approach to (dis)assembly sequencing, in: Proc. IEEE Int. Conf. Autom. Sci. Eng., 2009, pp. 286–291.

[20] X. Zhao, C. Li, Y. Tang, J. Cui, Reinforcement learning-based selective disassembly sequence planning for the end-of-life products with structure uncertainty, IEEE Robot. Autom. Lett. 6 (4) (2021) 7807–7814.

[21] T. Kiyokawa, J. Takamatsu, T. Ogasawara, Assembly sequences based on multiple criteria against products with deformable parts, in: Proc. IEEE Int. Conf. Robot. Autom., 2021, pp. 975–981.

[22] Y. Laili, X. Li, Y. Wang, L. Ren, X. Wang, Robotic disassembly sequence planning with backup actions, IEEE Trans. Autom. Sci. Eng. 19 (3) (2022) 2095–2107.

[23] S. Dorn, N. Wolpert, E. Schömer, An assembly sequence planning framework for complex data using general voronoi diagram, in: Proc. IEEE Int. Conf. Robot. Autom., 2022, pp. 9896–9902.

[24] K. Wang, L. Gao, X. Li, P. Li, Energy-efficient robotic parallel disassembly sequence planning for end-of-life products, IEEE Trans. Autom. Sci. Eng. 19 (2) (2022) 1277–1285.

[25] Y. Tian, J. Xu, Y. Li, J. Luo, S. Sueda, H. Li, K.D. Willis, W. Matusik, Assemble them all: Physics-based planning for generalizable assembly by disassembly, ACM Trans. Graph. 41 (6) (2022).

[26] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197.

[27] T. Ebinger, S. Kaden, S. Thomas, R. Andre, N.M. Amato, U. Thomas, A general and flexible search framework for disassembly planning, in: Proc. IEEE Int. Conf. Robot. Autom., 2018, pp. 3548–3555.

[28] K.-M. Lee, M. Bailey-van Kuren, Modeling and supervisory control of a disassembly automation workcell based on blocking topology, IEEE Trans. Robot. Autom. 16 (1) (2000) 67–77.

[29] A. Cebulla, T. Asfour, T. Kröger, Speeding up assembly sequence planning through learning removability probabilities, in: Proc. IEEE Int. Conf. Robot. Autom., 2023, pp. 12388–12394.

[30] L. Ma, J. Gong, H. Xu, H. Chen, H. Zhao, W. Huang, G. Zhou, Planning assembly sequence with graph transformer, in: Proc. IEEE Int. Conf. Robot. Autom., 2023, pp. 12395–12401.

[31] Y. Ren, H. Jin, F. Zhao, T. Qu, L. Meng, C. Zhang, B. Zhang, G. Wang, J.W. Sutherland, A multiobjective disassembly planning for value recovery and energy conservation from end-of-life products, IEEE Trans. Autom. Sci. Eng. 18 (2) (2021) 791–803.

[32] P. Dario, M. Rucci, C. Guadagnini, C. Laschi, An investigation on a robot system for disassembly automation, in: Proc. IEEE Int. Conf. Robot. Autom., vol. 4, 1994, pp. 3515–3521.

[33] K. Hohm, H. Muller Hofstede, H. Tolle, Robot assisted disassembly of electronic devices, in: Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., vol. 2, 2000, pp. 1273–1278.

[34] E. Zussman, M.C. Zhou, Design and implementation of an adaptive process planner for disassembly processes, IEEE Trans. Robot. Autom. 16 (2) (2000) 171–179.

[35] D.-H. Kim, S.-J. Lim, D.-H. Lee, J.Y. Lee, C.-S. Han, A RRT-based motion planning of dual-arm robot for (dis)assembly tasks, in: Proc. IEEE Int. Symp. Robot, 2013, pp. 1–6.

[36] I. Rodrıguez, K. Nottensteiner, D. Leidner, M. Kaßecker, F. Stulp, A. Albu-Schäffer, Iteratively refined feasibility checks in robotic assembly sequence planning, IEEE Robot. Autom. Lett. 4 (2) (2019) 1416–1423.

[37] T. Bachmann, K. Nottensteiner, M.A. Roa, Automated planning of workcell layouts considering task sequences, in: Proc. IEEE Int. Conf. Robot. Autom., 2021, pp. 12662–12668.

[38] M. Atad, J. Feng, I. Rodríguez, M. Durner, R. Triebel, Efficient and feasible robotic assembly sequence planning via graph representation learning, in: Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2023.

[39] J. Liu, Z. Zhou, D.T. Pham, W. Xu, C. Ji, Q. Liu, Collaborative optimization of robotic disassembly sequence planning and robotic disassembly line balancing problem using improved discrete bees algorithm in remanufacturing, Robot. Comput.-Integr. Manuf. 61 (2020) 101829.

[40] J. Liu, Z. Xu, H. Xiong, Q. Lin, W. Xu, Z. Zhou, Digital twin-driven robotic disassembly sequence dynamic planning under uncertain missing condition, IEEE Trans. Ind. Info. 19 (12) (2023) 11846–11855.

[41] K. Zakka, A. Zeng, J. Lee, S. Song, Form2Fit: Learning shape priors for generalizable assembly from disassembly, in: Proc. IEEE Int. Conf. Robot. Autom., 2020, pp. 9404–9410.

[42] Y. Lee, E.S. Hu, J.J. Lim, IKEA furniture assembly environment for long-horizon complex manipulation tasks, in: Proc. IEEE Int. Conf. Robot. Autom., 2021, pp. 6343–6349.

[43] O. Aslan, B. Bolat, B. Bal, T. Tumer, E. Sahin, S. Kalkan, AssembleRL: Learning to assemble furniture from their point clouds, in: Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2022, pp. 2748–2753.

[44] M. Qu, Y. Wang, D.T. Pham, Robotic disassembly task training and skill transfer using reinforcement learning, IEEE Trans. Ind. Info. 19 (11) (2023) 10934–10943.

[45] J. Borràs, R. Heudorfer, S. Rader, P. Kaiser, T. Asfour, The KIT swiss knife gripper for disassembly tasks: A multi-functional gripper for bimanual manipulation with a single arm, in: Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2018, pp. 4590–4597.

[46] C. Klas, F. Hundhausen, J. Gao, C.R.G. Dreher, S. Reither, Y. Zhou, T. Asfour, The KIT gripper: A multi-functional gripper for disassembly tasks, in: Proc. IEEE Int. Conf. Robot. Autom., 2021, pp. 715–721.

[47] M. Wurster, M. Michel, M.C. May, A. Kuhnle, N. Stricker, G. Lanza, Modelling and condition-based control of a flexible and hybrid disassembly system with manual and autonomous workstations using reinforcement learning, J. Intell. Manuf. 33 (2022).

[48] Y. Zhang, H. Zhang, Z. Wang, S. Zhang, H. Li, M. Chen, Development of an autonomous, explainable, robust robotic system for electric vehicle battery disassembly, in: Proc. IEEE/ASME Int. Conf. Adv. Intell. Mech., 2023, pp. 409–414.

[49] G. Gorjup, G. Gao, A. Dwivedi, M. Liarokapis, Combining compliance control, CAD based localization, and a multi-modal gripper for rapid and robust programming of assembly tasks, in: Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2020, pp. 9064–9071.

[50] R. Prévost, E. Whiting, S. Lefebvre, O. Sorkine-Hornung, Make it stand: Balancing shapes for 3D fabrication, ACM Trans. Graph. 32 (4) (2013).

[51] K. Harada, S. Kajita, K. Kaneko, H. Hirukawa, ZMP analysis for arm/leg coordination, in: Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., vol. 1, 2003, pp. 75–81.

[52] T. Kiyokawa, T. Sakuma, J. Takamatsu, T. Ogasawara, Soft-jig-driven assembly operations, in: Proc. IEEE Int. Conf. Robot. Autom., 2021, pp. 3466–3472.

[53] J. Blank, K. Deb, P.C. Roy, Investigating the normalization procedure of NSGA-III, in: Proc. Evolutionary Multi-Criterion Optimization, 2019, pp. 229–240.

[54] T. Kiyokawa, N. Shirakura, Z. Wang, N. Yamanobe, I.G. Ramirez-Alpizar, W. Wan, K. Harada, Difficulty and complexity definitions for assembly task allocation and assignment in human–robot collaborations: A review, Robot. Comput.-Integr. Manuf. 84 (2023) 102598.

[55] T. Yoshikawa, Y. Yokokohji, Y. Yu, Assembly planning operation strategies based on the degree of constraint, in: Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 1991, pp. 682–687.

[56] Industrial Robotics Category Assembly Challenge Rules and Regulations 2018, The Industrial Robotics Competition Committee, 2018, [Online]. Available: https://worldrobotsummit.org/download/rulebook-en/rulebook-Assembly_Challenge.pdf.

[57] Z. Hu, W. Wan, K. Koyama, K. Harada, A mechanical screwing tool for parallel grippers—Design, optimization, and manipulation policies, IEEE Trans. Robot. 38 (2) (2022) 1139–1159.

[58] W. Wan, K. Harada, F. Kanehiro, Planning grasps with suction cups and parallel grippers using superimposed segmentation of object meshes, IEEE Trans. Robot. 37 (1) (2021) 166–184.

[59] J. Kuffner, S. LaValle, RRT-connect: An efficient approach to single-query path planning, in: Proc. IEEE Int. Conf. Robot. Autom., vol. 2, 2000, pp. 995–1001.

[60] R. Diankov, Automated Construction of Robotic Manipulation Programs (Ph.D. thesis), Carnegie Mellon University, Robotics Institute, 2010.

[61] T. Sakuma, T. Kiyokawa, J. Takamatsu, T. Wada, T. Ogasawara, Soft-Jig: A flexible sensing jig for simultaneously fixing and estimating orientation of assembly parts, in: Proc. IEEE Int. Conf. Robot. Autom., 2022, pp. 10945–10950.

[62] T. Sakuma, T. Kiyokawa, T. Matsubara, J. Takamatsu, T. Wada, T. Ogasawara, Jamming gripper-inspired soft jig for perceptive parts fixing, IEEE Access 11 (2023) 62187–62199.

[63] J.J. Park, P. Florence, J. Straub, R. Newcombe, S. Lovegrove, DeepSDF: Learning continuous signed distance functions for shape representation, in: Proc. IEEE/CVF Conf. Comput. Vis. Patern. Recognit., 2019, pp. 165–174.

[64] Z. Hao, H. Averbuch-Elor, N. Snavely, S. Belongie, DualSDF: Semantic shape manipulation using a two-level representation, in: Proc. IEEE/CVF Conf. Comput. Vis. Patern. Recognit., 2020, pp. 7628–7638.

[65] C.R. Qi, H. Su, K. Mo, L.J. Guibas, PointNet: Deep learning on point sets for 3D classification and segmentation, in: Proc. IEEE/CVF Conf. Comput. Vis. Pattern. Recognit., 2017, pp. 652–660.

[66] C.R. Qi, L. Yi, H. Su, L.J. Guibas, PointNet++: Deep hierarchical feature learning on point sets in a metric space, in: Proc. Adv. Neural Inf. Process. Syst., 2017, pp. 5105–5114.

[67] G. Qian, Y. Li, H. Peng, J. Mai, H. Hammoud, M. Elhoseiny, B. Ghanem, PointNeXt: Revisiting PointNet++ with improved training and scaling strategies, in: Proc. Adv. Neural Inf. Process. Syst., 2022.

[68] H. Seada, K. Deb, A unified evolutionary optimization procedure for single, multiple, and many objectives, IEEE Trans. Evol. Comput. 20 (3) (2016) 358–369.

[69] Y. Vesikar, K. Deb, J. Blank, Reference point based NSGA-III for preferred solutions, in: Proc. IEEE Symp. Ser. Comput. Intell., 2018, pp. 1587–1594.

[70] Í. nigo Elguea-Aguinaco, A. Serrano-Muñoz, D. Chrysostomou, I. Inziarte-Hidalgo, S. Bøgh, N. Arana-Arexolaleiba, A review on reinforcement learning for contact-rich robotic manipulation tasks, Robot. Comput.-Integr. Manuf. 81 (2023) 102517.

[71] E. Coronado, T. Kiyokawa, G.A.G. Ricardez, I.G. Ramirez-Alpizar, G. Venture, N. Yamanobe, Evaluating quality in human-robot interaction: a systematic search and classification of performance and human-centered factors, measures and metrics towards an Industry 5.0, J. Manuf. Syst. 63 (2022) 392–410.

[72] K. Takata, T. Kiyokawa, I.G. Ramirez-Alpizar, N. Yamanobe, W. Wan, K. Harada, Efficient task/motion planning for a dual-arm robot from language instructions and cooking images, in: Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2022, pp. 12058–12065.

[73] M.S. Sakib, D. Paulius, Y. Sun, Approximate task tree retrieval in a knowledge network for robotic cooking, IEEE Robot. Autom. Lett. 7 (4) (2022) 11492–11499.

[74] K. Takata, T. Kiyokawa, I.G. Ramirez-Alpizar, N. Yamanobe, W. Wan, K. Harada, Graph based framework on bimanual manipulation planning from cooking recipe, Robotics 11 (6) (2022).

[75] Z. Wang, T. Kiyokawa, I. Sera, N. Yamanobe, W. Wan, K. Harada, Error correction in robotic assembly planning from graphical instruction manuals, IEEE Access 11 (2023) 107276–107286.

Takuya Kiyokawa received the B.E. degree from the National Institute of Technology, Kumamoto College, Japan, and the M.E. degree and the Ph.D. degree in engineering from the Nara Institute of Science and Technology, Japan, in 2018 and 2021, respectively. From 2021 to 2022, he was with the University of Osaka, Japan, as a Specially-Appointed Assistant Professor, and with the Nara Institute of Science and Technology, as a Specially-Appointed Assistant Professor. From 2023 to 2024, he was a Visiting Researcher with the Institute of Robotics and Mechatronics, German Aerospace Center (DLR), Oberpfaffenhofen, Weßling, Germany and since 2023, he has been with the University of Osaka, as an Assistant Professor. His current research interests include robot manipulation and agile reconfigurable robotic systems.

Kensuke Harada received the B.Sc., M.Sc., and Ph.D. degrees in mechanical engineering from Kyoto University, Kyoto, Japan, in 1992, 1994, and 1997, respectively. From 1997 to 2002, he was a Research Associate with Hiroshima University, Hiroshima, Japan. Since 2002, he has been with the National Institute of Advanced Industrial Science and Technology (AIST). From 2005 to 2006, he was a Visiting Scholar with the Department of Computer Science, Stanford University, Stanford, CA, USA, and the Leader of the Manipulation Research Group, AIST, from 2013 to 2015. He is currently a Professor with the Graduate School of Engineering Science, the University of Osaka, Toyonaka, Japan. His research interests include mechanics and control of robot manipulators and robot hands, biped locomotion, and motion planning of robotic systems.

Weiwei Wan received the Ph.D. degree in robotics from the Department of Mechano-Informatics, The University of Tokyo, Tokyo, Japan, in 2013. From 2013 to 2015, he was a Postdoctoral Researcher with Carnegie Mellon University, Pittsburgh, PA, USA, under the support of the Japan Society for the Promotion of Science (JSPS). From 2015 to 2017, he was a tenure-track Research Scientist with the National Institute of Advanced Industrial Science and Technology (AIST), Tsukuba, Japan. He is currently an Associate Professor with the Graduate School of Engineering Science, the University of Osaka, Toyonaka, Japan. His research interests include robotic manipulation and smart manufacturing.

Tomoki Ishikura obtained his master's degree from Nara Institute of Science and Technology in 2021. Since joining the company, he has been engaged in the development of an automated disassembly system for used home appliances at a recycling facility. Recently, he has focused on refurbishment as a circular economy business and is working on developing an autonomous disassembly system using AI and robotics technology to revolutionize the resource circulation system centered on crushing.

Naoya Miyaji conducted research on methodologies for product disassemblability design during his studies at the University of Osaka. Since joining Panasonic in 2015, he has been responsible for the development of home appliance recycling technology. Recently, he has focused on refurbishment as a circular economy business and is working on developing an autonomous disassembly system using AI and robotics technology to revolutionize the resource circulation system centered on crushing.

Genichiro Matsuda is a manager of resource circulation technology development in the R&D department of Panasonic Holdings. He has been in charge of the development of home appliance recycling technology since 2015. Toward the realization of a sustainable society, he has promoted the development of recycling technology for cellulose composite plastics and the development of resource efficiency indicators for circular economy products and services. In recent years, he has focused on refurbishment as a circular economy business, and is developing an autonomous disassembly system that utilizes AI and robotics technology to revolutionize the resource circulation system centered on crushing.