Robotic disassembly sequence planning with backup actions

Laili, Yuanjun; Li, Xiang; Wang, Yongjing; Ren, Lei; Wang, Xiaokang

DOI:

[10.1109/TASE.2021.3072663](https://doi.org/10.1109/TASE.2021.3072663)

License:

Other (please specify with Rights Statement)

Document Version

Peer reviewed version

Citation for published version (Harvard):

Laili, Y, Li, X, Wang, Y, Ren, L & Wang, X 2021, 'Robotic disassembly sequence planning with backup actions', *IEEE Transactions on Automation Science and Engineering*. <https://doi.org/10.1109/TASE.2021.3072663>

[Link to publication on Research at Birmingham portal](#)

Publisher Rights Statement:

© 2021 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

Y. Laili, X. Li, Y. Wang, L. Ren and X. Wang, "Robotic Disassembly Sequence Planning With Backup Actions," in *IEEE Transactions on Automation Science and Engineering*, doi: 10.1109/TASE.2021.3072663.

General rights

Unless a licence is specified above, all rights (including copyright and moral rights) in this document are retained by the authors and/or the copyright holders. The express permission of the copyright holder must be obtained for any use of this material other than for purposes permitted by law.

- Users may freely distribute the URL that is used to identify this publication.
- Users may download and/or print one copy of the publication from the University of Birmingham research portal for the purpose of private study or non-commercial research.
- User may use extracts from the document in line with the concept of 'fair dealing' under the Copyright, Designs and Patents Act 1988 (?)
- Users may not further distribute the material nor use it for the purposes of commercial gain.

Where a licence is displayed above, please note the terms and conditions of the licence govern your use of this document.

When citing, please reference the published version.

Take down policy

While the University of Birmingham exercises care and attention in making items available there are rare occasions when an item has been uploaded in error or has been deemed to be commercially or otherwise sensitive.

If you believe that this is the case for this document, please contact UBIRA@lists.bham.ac.uk providing details and we will remove access to the work immediately and investigate.

Robotic Disassembly Sequence Planning with Backup Actions

Yuanjun Laili, *Member, IEEE*, Xiang Li, Yongjing Wang, Lei Ren, *Member, IEEE*, and Xiaokang Wang

Abstract—A key step in remanufacturing is disassembly of the ‘core’, or the returned product to be remanufactured. Disassembly sequence planning is challenging, due to uncertainties in the conditions of the cores. Rust, corrosion, deformation and missing parts may require disassembly plans to be changed and adapted frequently. Conventional industrial automation that usually serves in repetitive and structured activities may fail when it is applied to disassembly. This research investigates the flexible sequencing of robotic disassembly in the presence of failed automation operations and develops online recovery by incorporating backup actions. It starts with modeling the time and success rate of a backup action. The expected disassembly time and completion rate of a disassembly plan are deduced according to the failure probability of both the operations and their backup actions. A bi-objective optimization model for robotic disassembly sequence planning is established using a dual-selection multi-objective evolutionary algorithm. Two solution selection criteria are combined to produce potential offspring candidates in each evolutionary generation. Experimental results show that the backup actions allow efficient recovery from automation and can potentially improve the robustness of robotic disassembly.

Note to Practitioners—This research was motivated by the development of automated disassembly techniques. Industrial automation techniques usually use pre-determined operation motions. Robotic disassembly using such an approach may fail due to uncertainties in the condition of the products (e.g. positioning and geometry). This paper introduces backup actions for disassembly sequence planning and describes the logic and reasoning of their implementation. Our proposed method can theoretically increase the completion rate of automated robotic disassembly. Experimental studies suggested that backup actions are efficient in providing a reliable disassembly sequence, and thus can improve the robustness of robotic disassembly. In future research, we will implement typical backup actions and establish an automated disassembly process with a re-planning module.

Index Terms—robotic disassembly; sequence planning; backup actions; multi-objective evolutionary algorithm

I. INTRODUCTION

Autonomous disassembly using robots has been proposed to replace costly and inefficient manual disassembly [1]. It requires the automatic generation of efficient disassembly plans, which describe the sequence of parts during the disassembly [2], [3].

Y. Laili, X. Li and L. Ren are with the School of Automation Science and Electrical Engineering, Beihang University, Beijing, 100191, China. (e-mail: lailiyuanjun@buaa.edu.cn, lllixxxiang@buaa.edu.cn, renlei@buaa.edu.cn)

Y. Wang is with the School of Engineering, University of Birmingham, Birmingham, B15 2TT, UK. (e-mail: y.wang@bham.ac.uk)

X. Wang is with the Department of Computer Science, St. Francis Xavier University, NS, B2G 2W5, Canada. (e-mail: xkwang@stfx.ca)

Y. Wang is the corresponding author (e-mail: y.wang@bham.ac.uk).

The end-of-life (EOL) products’ complex and unknown condition requires disassembly sequence plan to be flexible. However, current industrial automation systems are designed mostly for pre-determined and repetitive tasks and cannot deal with such a high-level of uncertainty [4].

Some aspects of uncertainty in EOL products and their effects on disassembly sequence planning have been studied in the literature. For example, disassembly time was seen as a normal distribution or a fuzzy system [2], [5], and a product’s condition can be recognized through feature identifications [6]. Human-robot collaboration in disassembly (i.e. cobotic disassembly) was also proposed to deal with uncertainties [7].

Uncertainties (e.g. rusted bolts and jammed bearings) may also cause a robotic disassembly operation to fail. The issue of failure has been discussed in disassembly line balancing problem where preventive and destructive strategies were proposed [8]. The research assumed the failures were repairable or skippable and a sequence plan could always be executed successfully.

In practice, a predetermined sequence plan can be easy to fail due to uncertainties in EOL products. Such a lack of robustness makes the implementation of disassembly automation difficult. The motivation of this research is to identify a strategic approach to improve the robustness of disassembly sequence planning.

There are three common actions to remove a component when the pre-assigned operation fails: changing the disassembly direction, adjusting the pre-assigned operation with an auxiliary action such as vibration, and changing the disassembly tool. For instance, the method to remove a jammed bearing include pulling from different direction, pulling while applying vibration, and using a different tool. These actions perform the role of backup in disassembly processes. If the pre-determined operation fails, one or a combination of these operations can be triggered, and increase the completion rate of a sequence plan.

Therefore, our research studies failure in robotic disassembly sequence planning, and proposes to use three types of backup actions to deal with disassembly failures (e.g. unable to remove a jammed bolt). Backup actions are the additional operations following the failed removal of a component.

The main contribution of this paper are as follows.

- (1) This paper proposes to use backup actions to deal with disassembly failures. Three possible backup actions are introduced.
- (2) This paper re-models the optimization of robotic disassembly sequence planning by incorporating backup actions.

(3) A dual-selection multi-objective evolutionary algorithm, DS-MOEA, is proposed to solve this disassembly sequence planning problem. It combines two selection rules from the non-dominated sorting genetic algorithm III (NSGA-III) [9] and an indicator-based evolutionary algorithm (IBEA) [10] to find more evenly distributed non-dominated solutions. This paper also modifies the precedence preservation crossover (PPX) proposed in [11] and provides a problem-specific mutation to update the individuals. Three case studies are given to verify the planning model and the algorithm. Experimental results show that backup actions improve the expected completion rate greatly. The proposed algorithm outperforms four other classical algorithms by identifying better sequence plans with much higher completion rate.

The remaining sections of this paper are as follows. Section II provides a brief overview of robotic disassembly sequence planning. Section III describes a re-formulated mathematical model of robotic disassembly sequence planning, considering three types of backup actions. The structure and operators of the DS-MOEA are introduced in Section IV. Then section V demonstrates the use of backup actions and the proposed DS-MOEA on three case studies. Finally, some conclusions are given in Section VI.

II. RELATED WORKS

This section briefly introduces state-of-the-art research on robotic disassembly sequence planning.

A. Mathematical Representation of Robotic Disassembly Sequence Planning

A variety of mathematical representations have been developed to model the relationship of a product's components, including the AND/OR graph, disassembly tree and interference matrices [12], [13]. In the context of using robots in disassembly, sequence planning concerns not only the disassembly sequence of the components and subassemblies [14], but also the disassembly directions [15]. An interference matrix is a popular tool for robotic disassembly planning, due to its capability of providing information concerning the directions.

Among the studies using interference matrices, many have assumed that both the product structure and the time to remove each component are fixed [16], [17]. Viewing disassembly time as a variable also has its place in the literature. For example, the component recovery type and material type have been seen as two factors that determine the time of a disassembly operation [18]. Different ways of calculating disassembly time, cost and recovery value in the case of EOL vehicles have been proposed to evaluate the dismantlability of a component [19].

Studies on disassembly sequence planning can also focus on the economic and environmental impacts [20]–[23]. Since the direction of the disassembly is less a concern in those topics, other types of mathematical representations may be adopted.

B. Methods for Robotic Disassembly Sequence Planning

Methods for robotic disassembly sequence planning include mathematical programming algorithms, heuristics and meta-heuristics [16], [21], [24], [25]. Some previous research has

assumed that the disassembly sequence can be the reverse of the assembly sequence. Hence, methods designed for assembly processes have been adapted to disassembly [26]. Ghandi et al. [3] summarized eight types of methods that can be used for simple assembly and disassembly sequence planning.

One of the most commonly used methods is a genetic algorithm (GA) based on PPX. Research focusing on designing better optimization algorithms [27], [28] tends to ignore the possible failure caused by uncertain product conditions.

To implement automated disassembly, a vision system has been introduced to detect detachable components to guide sequence planning online [29]. A majority of the above sequence planning methods assume that the condition of a product is known. It is difficult to apply the methods when the component interference relationships are uncertain due to rust, corrosion and deformation.

There are pioneering attempts to deal with uncertainties in the returned products by integrating stochastic simulation in a GA for disassembly sequence planning [30]. However, this method was designed from the perspective of economic return. The technical method to deal with uncertainties and automation failure has still not been developed.

In summary, existing research on disassembly sequence planning tends to use the following assumptions: (1) all components can be removed without failure; (2) the disassembly time of removing each component is pre-determined without change; (3) the transition time between two components is determined by the product structure and the preassigned disassembly tool.

The uncertainty of disassembly is not reflected in the above assumptions. The method for increase the completion rate of a disassembly sequence plan has not been established.

III. PROBLEM FORMULATION

The aim of this research is to incorporate backup actions in disassembly sequence planning to deal with failures. The main notations in the rest of this paper are defined in Table I.

A. Problem Preliminaries

Information about the direction of disassembly is contained in an interference matrix. An interference matrix can represent a product as defined in Eq. (1).

$$\mathbf{I} = \begin{bmatrix} \mathbf{0} & \mathbf{I}_{12} & \cdots & \mathbf{I}_{1n} \\ \mathbf{I}_{21} & \mathbf{0} & \cdots & \mathbf{I}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{I}_{n1} & \mathbf{I}_{n2} & \cdots & \mathbf{0} \end{bmatrix}, \quad (1)$$

where $\mathbf{I}_{ij} = [I_{ij1}, I_{ij2}, \dots, I_{ijD}]$ is a binary vector describing whether component j obstructs the removal of component i along D directions. If $I_{ijk}, k \in [1, D]$ is 1, it means component j is an obstruction for component i in direction k . Otherwise, it is 0. n refers to the number of components. The diagonals of $\mathbf{I}$ are all zero since a component does not obstruct itself.

An interference matrix provides information about the available disassembly directions of each component. This is the main reason that this research paper adopts interference

TABLE I
THE MAIN NOTATIONS DEFINED IN THIS PAPER

Notations	Matrices and sets
I	Interference matrix of a product
S	Layered candidate set [31] involving detachable components and subassemblies
d_i	List of directions from which E_i can be removed
b	Boundary list of S . It records the endpoints of layers.
P	A population of a MOEA
U	The Pareto set maintained in a MOEA
F_i	The i th non-dominated level of P using the fast non-dominated sorting strategy
Notations	Parameters
D	Number of disassembly directions
E_i	The i th detachable element in S . It can be either a component or a subassembly.
m	Number of candidate elements in S
l	Number of layers in S
$q_{x_i,k}$	The success rate of the k th backup action for removing x_i
$\tau_{x_i,k}$	The time for executing the k th backup action to remove x_i
t_{x_i}	Time taken in dismantling x_i (without backup action)
$\varphi_{y_i,y_{i+1}}$	Direction change time from y_i to y_{i+1}
ν_{x_i}	The velocity of dismantling x_i by the original operation
ν'_{x_i}	The velocity of dismantling x_i by an auxiliary recovery action, i.e., the second type of backup action
$\chi_{x_i,2}$	The setup time of dismantling x_i by an auxiliary recovery action, i.e., the second type of backup action
λ_i	The tool for removing x_i
$\phi_{\lambda_i,\lambda'_i}$	The tool change time from an old tool λ_i to a new tool λ'_i
ζ_{x_i,λ'_i}	The operational time of removing x_i by λ'_i
Λ	A binary label to record the actual recovery result of the k th action in a disassembly process
Δ	An inertia to accept new experience for updating $q_{x_i,k}$
r_{x_i}	Failure probability of the original operation for removing x_i
p_{x_i}	Failure probability of disassembling x_i when a backup action is applied
t_{x_i}	Expected time of disassembling x_i
n	Size of a MOEA's population
p_m	Probability of a mutation operator
T	A sequence plan's total time
Υ	A sequence plan's completion rate
Notations	Decision variables
x_i	The i th element to be removed in a disassembly sequence
y_i	The selected directions from which x_i is removed
$z_{x_i,k}$	A flag determines whether the k th backup action is prepared for x_i

matrix rather than graph representations, such as AND/OR graph (which do not include direction information and require manual extraction).

A method to extract removable components and subassemblies is the two-pointer detection strategy. [31]. The method finds components having no obstruction by Eq. (2).

Fig. 1. An example product with a undetachable component F4

$$\sum_{j=1}^n I_{ij}d_j = 0 \quad (2)$$

When multiple components interfere with one other, the two-pointer detection strategy [31] finds subassemblies (formulated by interlocked components) by allocating the start point and the end point of the interconnected components.

Take the product shown in Fig. 1 as an example, one assumes that the fastener F4 is undetachable due to corrosion. Its interference matrix can be shown in the right side of Fig. 1. The process of the two-pointer detection strategy for finding possible subassemblies can be shown in Fig. 2. It uses a forward pointer k and a backward pointer q to find dependent components that can be removed in combination. In each step of the detection, k moves to the component which impedes the current one, while q moves back to the earliest dependent component which impedes the component pointed by k . Details of the strategy can be found in [31].

In each step, detected components and subassemblies form a candidate subset, and removed from the interference matrix. The next step performs the same detection method to generate a new candidate subset. This step is performed in iterations until all components are removed from the interference matrix. The candidate subsets generated in different iterations form a structure of multiple layers, termed a layered candidate set S . Denoting the number of elements to be disassembled as $m = |S|$ and the number of layers (i.e., subsets) as l , a layered candidate set can be expressed by Eq. (3).

$$\begin{aligned}
 S &= \{S_1 | S_2 | \dots | S_l\} \\
 &= \{E_1 \circ d_1, \dots, E_{b_1} \circ d_{b_1} | \\
 &\quad E_{b_1+1} \circ d_{b_1+1}, \dots, E_{b_2} \circ d_{b_2} | \dots | \\
 &\quad E_{b_{l-1}+1} \circ d_{b_{l-1}+1}, \dots, E_m \circ d_m\},
 \end{aligned} \quad (3)$$

where $S_i = \{E_{b_{i-1}+1} \circ d_{b_{i-1}+1}, \dots, E_{b_i} \circ d_{b_i}\}$, $j \in [1, l]$ is a layer representing a group of detachable elements. E_i represents a component or a subassembly, while the list d_i defines the available directions from which E_i can be removed. The size of d_i satisfies $0 < |d_i| < D$. A boundary list $b = \{b_1, b_2, \dots, b_{l-1}\}$ is also introduced to record the endpoint of each layers.

Assigning a specific order to the elements in each layer will generate a complete sequence. An element in the disassembly sequence can be swapped only with those in the same layer. Two elements belonging to different layers are non-exchangeable. $S_j, j \in [1, l-1]$ is the predecessor of the target elements in $S_{j+1}, j \in [1, l-1]$. These rules ensure that

Fig. 2. The main procedure of the two point detection strategy for finding subassemblies of the example EOL product. (k is a forward point in each step, q is a backward pointer)

Fig. 3. The main procedure of the two point detection strategy for finding subassemblies of the example EOL product. (k is a forward point in each step, q is a backward pointer)

the sequence does not violate the precedence constraint defined in the interference matrix.

The layered candidate set of the example product generated by the two-pointer detection strategy can be illustrated by Fig. 3. It is represented as $\{F1 \circ [Y-], F2 \circ [X+] \mid \{C1, C3, F3\} \circ [X-, Y+], \{C2, C4, F4\} \circ [X+, Y-] \mid F3 \circ [X+] \mid C1 \circ [X+, Y-], C3 \circ [X-, Y+]\}$, which includes four layers. The first layer has two components, which can be disassembled from the direction $Y-$ and the direction $X+$, respectively. The second layer has two subassemblies, $\{C1, C3, F3\}$ and $\{C2, C4, F4\}$, which can be disassembled from two directions, $[X- \mid Y+]$ and $[X+ \mid Y-]$, respectively. The third layer has only one component $F3$, which can be removed from only the $X+$ direction; while the fourth has two components. According to the layered candidate set, there are eight ($2 \times 2 \times 1 \times 2 = 8$) possible disassembly plans in total.

The disassembly sequence planning problem is to find an optimal sequence $\{x_1, x_2, \dots, x_m\}$ and directions $\{y_1, y_2, \dots, y_m\}$ of removing the m elements in the candidate set S . The i th ($i < m$) element to be removed in the sequence is defined as x_i , while the operating direction for x_i is y_i .

B. Backup Actions

As a task failure can happen in any step of a disassembly sequence, every disassembly operation to remove element x_i in a disassembly sequence has a failure probability r_{x_i} .

Backup actions are additional actions to disassemble a target when a failure is detected. Each action requires a

certain amount of time and has a probability to recover a disassembly process from a failure. A robot does not have the intelligence to select the most appropriate backup action following a failure. Thus, we incorporate the selection of the backup actions with the aim to establish a reliable automated disassembly sequence plan.

Three types of backup actions are discussed in this paper, i.e. direction change action, auxiliary recovery action, and tool change action.

A direction change action performs the original operation in another direction, it requires a short time to adjust the direction of robotic manipulator and re-perform the disassembly operation. Suppose the basic disassembly time of x_i is t_{x_i} , and the direction change time from a direction y to y' is $\varphi_{y,y'}$, the time of a direction change action for handling the target x_i can be estimated according to Eq. (4).

$$\tau_{x_i,1} = t_{x_i} + \varphi_{y_i,y'_i} \quad (4)$$

An auxiliary recovery action performs an operation, such as applying vibration additional to the original operation. The action does not require changing robotic tools. The time of such an auxiliary recovery action is determined by its setup time and the EOL condition of the target component. For example, if a long shaft is jammed due to corrosion and rust, applying vibration while pulling at a low speed can help to extract the shaft. The time of this backup action is determined primarily by the velocity of the pulling. Let the original velocity of dismantling x_i to be ν_{x_i} and the new velocity (while an auxiliary recovery action is applied) to be ν'_{x_i} , the setup time of this action is $\chi_{x_i,2}$, the time of an auxiliary recovery action (i.e. the second type of back action) can be estimated by Eq. (5). The two velocities and the setup time are three pre-known parameters in the disassembly process.

$$\tau_{x_i,2} = \chi_{x_i,2} + t_{x_i} \cdot \frac{\nu_{x_i}}{\nu'_{x_i}} \quad (5)$$

A tool change action adopts a new tool to remove the target component. This may involve complex tasks, such as using heat gun, adopting hammer to loosen the target, and using cutter or drill. Therefore, this type of action can be non-destructive or destructive. The time of a tool change action for an element x_i is the summation of the time to change from an old tool λ_i to the new tool λ'_i , defined as $\phi_{\lambda_i,\lambda'_i}$, the operational time by the new tool, denoted as ζ_{x_i,λ'_i} , the time to change back to original tool $\phi_{\lambda'_i,\lambda_i} = \phi_{\lambda_i,\lambda'_i}$, and the basic operational time to remove the target. Since destructive is available in this type of action, its success rate is higher than the other two actions.

$$\tau_{x_i,3} = 2\phi_{\lambda_i,\lambda'_i} + \zeta_{x_i,\lambda'_i} + t_{x_i} \quad (6)$$

The success rate of a backup action can be obtained from practice empirically. One simple way is to set the success rate of the action as 1 initially, and update it according to the actual recovery result when task failure happens. Let the success rate of a backup action of type k for removing x_i to be $q_{x_i,k}$, it can be updated by Eq. (7).

$$q_{x_i,k}^{(new)} = \Delta q_{x_i,k}^{(old)} + (1 - \Delta)\Lambda, \quad (7)$$

where Λ is a binary label to denote the actual recovery result of the k th action in a disassembly process. $\Lambda = 1$ represents the target element is successfully disassembled by the backup action. Otherwise, $\Lambda = 0$. Δ is an inertia to accept new experience for updating $q_{x_i,k}$.

It should be noted that whether the first action is applicable to the i th element in S depends on its direction set d_i . If there is merely one possible direction in d_i to disassemble E_i , the direction change action would be invalid and thus the other two options should be chosen directly.

C. Assumptions

Since backup actions can be one of the above types or a series of operations including direction change, tool change and auxiliary adjustments, this paper only consider the simplest backup actions. To clarify the scope of this work, the following assumptions are made:

1. Each operation removes a component or a subassembly.
2. The basic disassembly operation is executed at a fixed time.
3. Tool change and direction change are performed automatically at a standard speed.
4. Each operation is carried out in linear movements along principal directions only.
5. Three simple backup actions, i.e., direction change action, auxiliary recovery action, and tool change action are predetermined for each possible disassembly step. They are applied in a fixed order.
6. If the basic disassembly operation and the three backup actions are all failed, the disassembly process will terminate.
7. Both the basic disassembly operation and the backup actions do not change the disassembly relationships of the remaining components.

Therefore, the model established in this paper is applicable only in the scenarios that a robotic manipulator disassembles an EoL product using pre-determined linear motions. The manipulator is limited into six degrees of freedom. The backup actions are also simplified as the above three short attempts. The complex cases which requires more than three backup actions and some combination actions are not considered in this paper. For example, the case that both a tool change action and some more auxiliary actions are required after attempting the first two backup actions is excluded from the model.

D. Expected Disassembly Time and Completion Rate

To incorporate the selection of backup actions as decision variables, this paper denotes $z_{i,k}, i \in [1, m], k \in [1, 3]$ as a binary variable of determining whether the k th backup action is adopted to disassemble the element x_i . For example, $z_i = \{z_{i,1}, z_{i,2}, z_{i,3}\} = 010$ means that only the auxiliary recovery action is adopted to remove x_i if a failure happens. When multiple backup actions are adopted, the three actions are applied sequentially. For instance, $z_i = \{z_{i,1}, z_{i,2}, z_{i,3}\} = 101$ means that the first and the third backup actions are planned.

When a failure happens, the first backup action is applied to remove x_i . If x_i is still unremovable, the third backup action will be applied.

The original failure probability to disassemble x_i has been defined as r_{x_i} . Let the success rate of the k th backup action for removing x_i as $q_{x_i,k}$, Eq. (8) defines the failure probability of disassembling x_i when backup actions are prepared.

$$p_{x_i} = r_{x_i} \prod_{k=1}^3 (1 - z_{i,k} q_{x_i,k}) \quad (8)$$

The basic time taken in dismantling x_i has been defined to t_{x_i} and the direction change time between two directions y_i and y_{i+1} is $\varphi_{y_i y_{i+1}}$. Metrics proposed by Kroll et al. can be used to estimate t_{x_i} [32]; while metrics proposed by Liu et al. can be used to estimate the transition time $\varphi_{y_i y_{i+1}}$ [15]. Marconi et al. propose analyzing historical data for these estimation [33]. Accordingly, the expected disassembly time $\bar{t}_{x_i}$ for x_i with backup actions can be calculated by using Eq. (9).

$$\bar{t}_{x_i} = \begin{cases} t_{x_i} + \varphi_{y_i, y_{i+1}} + r_{x_i} z_{i,1} \tau_{x_i,1} \\ \quad + r_{x_i} (1 - z_{i,1} q_{x_i,1}) z_{i,2} \tau_{x_i,2} \\ \quad + r_{x_i} (1 - z_{i,1} q_{x_i,1}) (1 - z_{i,2} q_{x_i,2}) z_{i,3} \tau_{x_i,3} \\ t_{x_i} + r_{x_i} z_{i,1} \tau_{x_i,1} \\ \quad + r_{x_i} (1 - z_{i,1} q_{x_i,1}) z_{i,2} \tau_{x_i,2} \\ \quad + r_{x_i} (1 - z_{i,1} q_{x_i,1}) (1 - z_{i,2} q_{x_i,2}) z_{i,3} \tau_{x_i,3} \end{cases} \quad \begin{matrix} \text{if } i < m \\ \text{if } i = m \end{matrix} \quad (9)$$

If applying the selected backup actions still cannot remove x_i , the disassembly process will terminate and request manual checks and remedy by a human worker. In this case, the overall disassembly time of a sequence plan upon the termination is formulated by Eq. (10).

$$\begin{aligned} T &= p_{x_1} \bar{t}_{x_1} + (1 - p_{x_1}) p_{x_2} \sum_{i=1}^2 \bar{t}_{x_i} + (1 - p_{x_1}) (1 \\ &\quad - p_{x_2}) p_{x_3} \sum_{i=1}^3 \bar{t}_{x_i} + \cdots + \prod_{j=1}^{m-2} (1 - p_{x_j}) p_{x_{m-1}} \\ &\quad \sum_{i=1}^{m-1} \bar{t}_{x_i} + \prod_{j=1}^{m-1} (1 - p_{x_j}) \sum_{i=1}^m \bar{t}_{x_i} \\ &= p_{x_1} \bar{t}_{x_1} + \sum_{s=2}^{m-1} \left(\prod_{j=1}^{s-1} (1 - p_{x_j}) p_{x_s} \sum_{i=1}^s \bar{t}_{x_i} \right) \\ &\quad + \prod_{j=1}^{m-1} (1 - p_{x_j}) \sum_{i=1}^m \bar{t}_{x_i} \end{aligned} \quad (10)$$

If a disassembly terminates at the first task x_1 , the overall disassembly time is $\bar{t}_1$ only. The probability that the process terminates at the i th ($1 < i < m$) task to disassemble x_i is $(1 - p_{x_1}) \cdots (1 - p_{x_{i-1}}) p_{x_i}$. The expected time to remove m elements is $\sum_{i=1}^m \bar{t}_{x_i}$.

Accordingly, the completion rate of the sequence plan is defined as Υ and can be calculated as in Eq. (11). The numerator represents the number of successful operations, and

the denominator is the total number of elements m to be disassembled.

$$\Upsilon = \left[p_{x_1} + \sum_{s=2}^{m-1} \left(s \prod_{j=1}^{s-1} (1 - p_{x_j}) p_{x_s} \right) + m \prod_{j=1}^{m-1} (1 - p_{x_j}) \right] / m \quad (11)$$

E. Mathematical Model

In summary, the robotic disassembly sequence planning problem with backup actions contains $m + m + 3m = 5m$ variables as follows.

(1) $\{x_1, x_2, \dots, x_m\}$ defines the sequence of the candidate elements to be disassembled. Each variable $x_i, i \in [1, m]$ is an integer range from 1 to m ;

(2) $\{y_1, y_2, \dots, y_m\}$ designates the direction to disassemble an element in the above sequence. Each variable $y_i, i \in [1, m]$ is an integer range from 1 to D ;

(3) $\{z_{x_1,1}, z_{x_1,2}, z_{x_1,3}, \dots, z_{x_m,1}, z_{x_m,2}, z_{x_m,3}\}$ determines whether the three types of backup actions are prepared for disassembling the elements in the above sequence. Each variable $z_{x_i,j}, i \in [1, m], j \in [1, 3]$ is a bool number.

The main objectives of this research are to minimize the total disassembly time and maximize the completion rate of a disassembly sequence.

$$\begin{cases} (1) \text{ Minimize } T \\ (2) \text{ Maximize } \Upsilon \end{cases} \quad (12)$$

subject to,

- (1) $1 < x_i < m$
- (2) $\forall i, j \in [1, m], i \neq j \Leftrightarrow x_i \neq x_j$
- (3) $\forall i \in [1, m], y_i \in \mathbf{d}_{x_i}$
- (4) $\forall i \in [1, m], z_{x_i,1} = 0, \text{ if } |\mathbf{d}_{x_i}| = 1$

In the above list of constraints, (1) and (2) confine the first m variables to be a sequence in the range of $[1, m]$. In constraints (3) and (4), $\mathbf{d}_{x_i}$ is the possible direction set in the disassembly of x_i . Constraint (3) ensures the directions are valid as defined by the layered candidate set $\mathbf{S}$. If only one direction is available, the size of the corresponding direction list is 1, i.e. $|\mathbf{d}_{x_i}| = 1$. In this case, the direction change action is unavailable. Thus, constraint (4) defines the optional backup actions depending on whether the element x_i can be removed from another direction.

IV. THE DUAL-SELECTION MULTI-OBJECTIVE EVOLUTIONARY ALGORITHM

According to the above analysis, the solutions with high completion rate requires a long disassembly time, while the ones with low completion rate will be terminated in a short time. By introducing the completion rate as an objective of the disassembly sequence planning problem, the solution space is broadened greatly. The challenge is that, the existing method cannot find evenly distributed Pareto solutions in a short time. To overcome this challenge, this paper proposes a DS-MOEA,

Fig. 4. The main procedure of the DS-MOEA. This paper combines the selection schemes of the NSGA-III and the IBEA which are in the dashed boxes.

which incorporates two selection schemes from the NSGA-III and the IBEA to obtain a group of diverse evolution candidates. Based on the individual structure shown in Eq. (12), the framework of the DS-MOEA is shown in Fig. 4.

The process uses three populations, P , P' and P'' , and a Pareto set U ; n offspring are generated in each generation. P represents a union set consisting of $2 \cdot n$ solutions (i.e. individuals); P' is a mating pool for parent individuals. It contains n potential and diverse individuals selected from P . P'' is the offspring population randomly selected from the parent individuals in P' . The n solutions of P'' are then used to update U by a domination check [34]. In this step, the maximum size of U is set as U_m . If $|U| > U_m$, the individual of the minimum crowd distance is removed from U . Then P' and P'' are combined to form the new union set in the next iteration.

A. The Dual-selection Strategy for Solution Exploration

The solution selection acts as the primary criteria dividing the state-of-the-art MOEAs into three categories: Pareto-based algorithm, decomposition-based algorithm and indicator-based algorithm. Decomposition-based algorithm maintains representative individuals along a group of weight vectors; while the other two find the evolutionary seeds from a union set. To preserve a higher individual diversity in each iteration, this paper combines the selection schemes of both the NSGA-III and the IBEA as shown in Fig. 4.

In the first-round selection, the population P is sorted to several non-dominated levels $\{F_1, F_2, \dots, F_i, \dots\}$ using the fast non-dominated sorting strategy. The individuals from the first l fronts, which satisfy $\sum_{i=1}^l |F_i| \leq \frac{n}{2}$, are selected in the mating pool. If $\sum_{i=1}^l |F_i| = \frac{n}{2}$, the first-round selection is finished directly. Otherwise, the individuals of F_{l+1} are normalized by an ideal point in line with a group of evenly

Fig. 5. The dual-selection process of the DS-MOEA

distributed weight vectors. A set of reference points are also introduced to define the perpendicular distances between the individuals and the reference lines. The individuals in F_{l+1} which associate the reference point with the smallest niche are selected to fill the mating pool P' until the size of $|P'|$ reaches $\frac{n}{2}$. The details of the adaptive normalization, association and niche-preservation operation are available in [9].

In the second-round selection, the ε -indicator is introduced to designate a scalar fitness value to each individual of P . The rest of the individuals of the higher ε -indicator are selected through a binary tournament process. To be specific, two individuals are picked arbitrarily at a time and the one with the better ε -indicator is put into P' . The mating pool P' will be filled in $\frac{n}{2}$ times from a random selection.

The dual-selection strategy is summarized in Fig. 5. $\frac{n}{2}$ individuals of P' are selected from the first $l+1$ front levels, while the rest $\frac{n}{2}$ individuals are selected by ε -indicator-based tournament selection. Thus, potential individuals that are not in the first fronts but with high ε -indicators can be selected to improve the diversity of the mating pool P' . The proposed selection process ensures that the potential individuals selected by using either evaluation criterion are embraced.

It is worth noting that Pareto non-dominated selection is always performed before the indicator-based selection. Otherwise, the association-based niche preservation operation will be carried out for a front level ranking far behind the original one, and thus potential individuals with either a low domination level or a better indicator will be neglected.

B. The Variation Operators for Solution Exploitation

According to Eqs. (10) and (11), the success rate of each disassembly step is highly dependent on the previous step. The sequence considering dependent failure probabilities cannot be separated as independent blocks and it is inappropriate to calculate the disassembly time and success rate of a single block. Hence, many efficient operators [26], [35] cannot be applied to the model proposed in this paper. This paper adopts a simple precedence preservation-based operator to find new solutions in each generation. As the sequence to be optimized is followed by the direction and backup variables, a modified PPX and a bisectional mutation operator are designed to change the order of a detachable element, its direction, and the backup action choices.

Fig. 6. An example of the modified PPX operator

A solution can be regarded as a matrix of three rows: the disassembly order, disassembly direction, and backup action choices. Each step of the PPX performs on a column. An example of the modified PPX is demonstrated in Fig. 6. It includes two steps of information exchange. In the first step, a group of m random numbers are generated. If a random number $r_k, k \in [1, m]$ is lower than 0.5, the leftmost column of parent 1 is adopted to the offspring individual as its k th column. Otherwise, the leftmost column of parent 2 is assigned as the k th column of the offspring. The corresponding disassembly element in both parents is then removed after the assignment. Not only are the precedence constraints maintained from this operator, but the directions and backup actions corresponding to each element in different parents are interchangeably inherited to the offspring. In the next step, two elements of the child individual are selected randomly, three random numbers $\{r'_1, r'_2, r'_3\}$ are generated to determine whether their backup actions are exchanged. If $r'_k, k \in [1, 3]$ is larger than 0.5, the backup action-related variables of the two elements are exchanged.

The bisectional mutation is designed as shown in Fig. 7. It uses two random numbers r_1 and r_2 to generate a two-layer decision. The first layer decides whether the sequence variation or the element-wise variation is to be performed. When $r_1 < 0.5$, the sequence variation operator will locate a layer of the candidate set that contains more than one detachable element, and mutate the sub-sequence by internal element exchange. Similar to the PPX operator, the internal element exchange includes the swapping of two elements, x_i and x_j , but also their directions and backup action choices, i.e., y_i and y_j as well as z_i and z_j . On the contrary, if the element-wise variation is adopted, the second random number r_2 provides equal probability to change either the direction or the backup action of the element. If $r_2 < 0.5$, the operator will find an element that can be removed from multiple directions and arbitrarily change its disassembly direction. Otherwise, the backup action of a randomly chosen element will be changed accordingly.

It should be noted that the modified PPX operator is performed n times with probability one in each iteration to generate n offspring individuals. The bisection mutation

Fig. 7. The process of the bisectonal mutation operator

TABLE II
PARAMETER VALUES USED IN EXPERIMENTS¹

Definition	Notation	Range	Unit
The basic disassembly time	t_i	[10, 30]	s
The direction change time	$\varphi_{i,j}$	[2, 5]	s
The tool change time	$\phi_{i,j}$	[10, 20]	s
The position adjustment time	$\psi_{i,j}$	[5, 10]	s
The failure probability	r_i	[0.01, 0.3]	/
The success rate of backup 1	q_1	[0.2, 0.4]	/
The success rate of backup 2	q_2	[0.4, 0.7]	/
The success rate of backup 3	q_3	[0.7, 0.9]	/
The additional time of backup 1	τ_1	[3, 5]	s
The additional time of backup 2	τ_1	[5, 10]	s
The additional time of backup 3	τ_1	[10, 15]	s

operator is performed with probability p_m to update these new individuals to build the offspring population P'' .

C. Time Complexity of The DS-MOEA

The proposed DS-MOEA applies four steps, i.e. the fast non-dominated sorting; association and niche-preservation-based individual selection; ε -indicator-based fitness assignment; and tournament selection, to select candidate individuals; and then adopts a modified PPX operator and a bisectonal mutation operator to produce new individuals.

The fast non-dominated sorting step requires $O(n^2)$ times, where n represents the number of individuals produced in each generation. The association and niche-preservation-based individual selection step requires $O(n/2) = O(n)$. As the ε -indicator-based fitness assignment step and the tournament selection step requires $\frac{n}{2} + \frac{n}{2}$ times, the time complexity of the dual-selection strategy is $O(n^2)$, which is the same as in the NSGA-III.

The modified PPX operator and the bisectonal mutation operator requires $O(n * m)$ and $O(n)$ respectively; where m represents the number of target elements, i.e. the components and subassemblies to be disassembled.

Thus, the overall time complexity of the proposed algorithm is $O(n^2)$. As half of the candidate individuals are selected

by a simpler procedure with a lower time complexity, this evolutionary process is faster than the NSGA-III.

V. EXPERIMENTS AND DISCUSSIONS

This section gives five case studies. The performance of the DS-MOEA is compared to four typical MOEAs.

A. Product Instances and Experimental Configurations

The products to be disassembled is given in Fig. 8. Two products, the 'product C' and the 'product B' containing 15 and 22 components respectively, are introduced from [31]. Three products are from the 'GRABCAD' library [36]. They are a worm gear, a motor reducer and a gear pump, containing 38, 40 and 47 components, respectively. They were selected as their level of complexity is suitable for applying disassembly sequence planning and backup actions. The first two products can be disassembled from six disassembly directions in total, $(X-, X+, Y-, Y+, Z-, Z+)$, while the last three products should be disassembled from five directions, $(X-, X+, Y-, Y+, Z+)$. The components of each product are labeled in Fig. 8. Their interference matrices can be obtained using the CAD models either automatically or manually. Parameter values used in the optimizations are given in Table II. By using the detection strategy in [31], the layered candidate sets representing the five products are given in Table III.

The DS-MOEA is compared to four typical MOEAs, i.e. the IBEA (Indicator-based Evolutionary Algorithm); the MOEA/D (Multi-Objective Evolutionary Algorithm based on Decomposition); the NSGA-II (Non-dominated Sorting Genetic Algorithm II); and the NSGA-III. The variation operations of the four typical MOEAs are replaced by the problem specific PPX operator and the bisectonal mutation operator proposed in this paper. Using the reference point generation strategy in the NSGA-III and the MOEA/D, this research defines the parameter $h = 9$. As the number of objectives to be solved is 2, the size of the population is $n = C_{h+2-1}^{2-1} = 10$. The mutation probability is $p_m = 0.15$. The maximum number of function evaluations is 10^4 . The maximum size of the Pareto set U is set as 20. Three metrics, i.e. HV (Hypervolume), IGD (Inverted Generational Distance) [37], and ε -indicator [38], [39] are adopted to evaluate the performance of the above algorithms. Each algorithm is executed 50 times independently and generates 50 independent non-dominated sets. The non-dominated sets of all algorithms are normalized together. Each objective value of a non-dominated solution is normalized to $[0, 1]$ according to the maximum objective value and the minimum value obtained by all these algorithms. The reference point for calculating the HV is $\{1.1, 1.1\}$. The ideal points to obtain the IGD are the non-dominated solutions selected from the solutions obtained by all of the five MOEAs in 50 runs.

All experiments are implemented by Xcode C++ and tested in a MAC OS X environment. The hardware configuration is 2.3-GHz Intel Core i7 CPU with 8 GB of 1.6-GHz DDR3 RAM.

¹The value ranges of failure probability and success rates are estimated and for the demonstration of backup actions. More accurate values can be obtained through statistical analysis in practice.

Fig. 8. Five products for experiments

TABLE III
THE LAYERED CANDIDATE SETS OF THE DETACHABLE ELEMENTS WITH THEIR DIRECTION LISTS FOR THE THREE PRODUCTS

Product	The layered candidate set	m
Product C	$\langle 1 \circ [6], 2 \circ [6], 7 \circ [3], 8 \circ [4], 11 \circ [5], \langle 9 \circ [3, 4], 12 \circ [5], \langle 13 \circ [5], \langle 14 \circ [5], \langle 15 \circ [5], \langle 10 \circ [5] \rangle \rangle \rangle \rangle$ $\langle \{3, 4\} \circ [1, 2, 3, 4, 6], \{5, 6\} \circ [1, 2, 3, 4, 5], \langle 4 \circ [5], 5 \circ [6], \langle 3 \circ [1, 2, 3, 4, 5, 6], 6 \circ [1, 2, 3, 4, 5, 6] \rangle \rangle \rangle$	17
Product B	$\langle 11 \circ [5], 12 \circ [5], 18 \circ [1], 19 \circ [5], 20 \circ [5], 21 \circ [5], 22 \circ [5], \langle 7 \circ [5], 8 \circ [5], 17 \circ [1], \langle 4 \circ [5], 15 \circ [2] \rangle \rangle \rangle$ $\langle 16 \circ [2, 3, 4, 5, 6], \langle 14 \circ [2, 3, 4, 5, 6], \langle 13 \circ [3, 4, 6], \langle 9 \circ [6], 10 \circ [6], \langle 2 \circ [5], 3 \circ [5], 5 \circ [6], 6 \circ [6] \rangle \rangle \rangle \rangle$ $\langle 1 \circ [1, 2, 3, 4, 5, 6] \rangle$	22
Worm gear	$\langle 1 \circ [5], 2 \circ [5], 3 \circ [5], 4 \circ [5], 5 \circ [5], 6 \circ [5], 7 \circ [5], 8 \circ [5], 9 \circ [1], 12 \circ [1], 13 \circ [1], 17 \circ [2], 18 \circ [2], 19 \circ [2], 20 \circ [2], 21 \circ [2], 22 \circ [4], 23 \circ [4], 24 \circ [4], 25 \circ [4], 26 \circ [4], 32 \circ [3], 33 \circ [3], 34 \circ [3], 35 \circ [3], 36 \circ [3], \langle 10 \circ [1], 16 \circ [2], 27 \circ [4], 31 \circ [3], \langle 11 \circ [1], 15 \circ [2], 28 \circ [4], \langle 29 \circ [4], \langle \{38, 14\} \circ [5], \langle 14 \circ [1, 2], 30 \circ [5], 38 \circ [1, 2], \langle 37 \circ [1, 2, 3, 4, 5] \rangle \rangle \rangle \rangle \rangle$	39
Motor reducer	$\langle 1 \circ [1], 2 \circ [1], 3 \circ [1], 4 \circ [1], 5 \circ [1], 6 \circ [1], 7 \circ [1], 8 \circ [1], 22 \circ [5], 23 \circ [5], 29 \circ [4], 30 \circ [5], 31 \circ [3], 40 \circ [2], \langle 10 \circ [1], 32 \circ [2], 33 \circ [2], 34 \circ [2], 35 \circ [2], 39 \circ [2], \langle 9 \circ [1], 38 \circ [2, 3, 4, 5], \langle 11 \circ [1], 20 \circ [1], 37 \circ [2], \langle 12 \circ [1], 13 \circ [1], 14 \circ [1], 15 \circ [1], 16 \circ [1], 17 \circ [1], 18 \circ [1], 19 \circ [1], \langle 21 \circ [1], \langle 24 \circ [1], \langle 25 \circ [1], \langle 26 \circ [1], \langle 27 \circ [1], \langle 28 \circ [1], \langle 36 \circ [1, 2, 3, 4, 5] \rangle \rangle \rangle \rangle \rangle \rangle$	40
Gear pump	$\langle 1 \circ [1], 2 \circ [1], 3 \circ [1], 4 \circ [1], 5 \circ [1], 6 \circ [1], 7 \circ [1], 8 \circ [1], 9 \circ [1], 10 \circ [1], 26 \circ [4], 27 \circ [4], 28 \circ [4], 29 \circ [4], 30 \circ [3], 31 \circ [3], 32 \circ [3], 33 \circ [3], 39 \circ [2], 40 \circ [2], 41 \circ [2], 42 \circ [2], 43 \circ [2], 44 \circ [2], 45 \circ [2], 46 \circ [2], 47 \circ [2], \langle 11 \circ [1], 12 \circ [1], 24 \circ [1, 2, 4, 5], 25 \circ [1, 2, 3, 5], 37 \circ [2], \langle 13 \circ [1, 3, 4, 5], 22 \circ [1, 2, 4, 5], 23 \circ [1, 2, 3, 5] \rangle \rangle \rangle$ $\langle 16 \circ [1], 17 \circ [1], \langle 20 \circ [1], 21 \circ [1], \langle 18 \circ [2], 19 \circ [1, 2], \langle 38 \circ [2], \langle 36 \circ [1, 2], \langle 35 \circ [2, 3, 4, 5] \rangle \rangle \rangle \rangle \rangle$ $\langle 34 \circ [2, 3, 4, 5], \langle 15 \circ [2, 3, 4, 5], \langle 14 \circ [1, 2, 3, 4, 5] \rangle \rangle \rangle$	47

(a) The product B

(b) The product C

(c) The worm gear

(d) The motor reducer

(e) The gear pump

Fig. 9. The comparison of ideal disassembly time and expected disassembly time

Fig. 10. Experiment results of three case studies

TABLE IV
THE RANGE OF SOLUTIONS

Product	Backup	Time (s)	Completion rate (%)
'Product C'	×	[26.39, 27.17]	[40.45, 43.26]
	✓	[29.15, 27.36]	[40.45, 90.64]
'Product B'	×	[20.77, 22.19]	[35.31, 39.63]
	✓	[23.27, 22.57]	[35.42, 89.07]
Worm gear	×	[20.44, 23.65]	[17.11, 28.35]
	✓	[23.90, 23.86]	[20.10, 82.08]
Motor reducer	×	[22.50, 24.74]	[18.25, 25.34]
	✓	[22.86, 25.21]	[21.61, 81.46]
Gear pump	×	[18.76, 22.06]	[15.34, 27.81]
	✓	[21.35, 22.15]	[18.10, 81.01]

B. How Does the New Model Change the Calculation of Disassembly Time?

A main difference between this research and other works is the consideration of failures. If failure is not considered, disassembly time is the summation of operational times (i.e. times taken by the disassembly operation, direction change, tool change, etc.).

Here, the five case studies are used to calculate disassembly time without considering failures (i.e. ideal disassembly time) and the method proposed in this paper (i.e. expected disassembly time). Solutions are given in Fig. 9. In each case, we sample the 20 non-dominated solutions obtained by the DS-MOEA in an independent run. Hence, there are 1000 non-dominated solutions obtained by 50 independent runs.

An ideal time is longer than an expected time. This is because an ideal time is the time of a complete disassembly. An expected time can reflect the possibility of failures in which case disassembly may be terminated before all components are removed.

It is also worth noting that the expected times in all five cases vary over a large range; while the ideal times fluctuate over a smaller range. Hence, the advantage of identifying an optimized solution can be more obvious in the case of the proposed model.

C. The Effectiveness of Backup Actions

A solution's performance is determined by not only its expected time, but also the completion rate. The experiments collect two groups of non-dominated solutions obtained by the DS-MOEA in 50 independent runs with and without backup actions, respectively. The distributions of these non-dominated solutions are shown in Fig. 10. The yellow points and blue points represent the solutions without and with backup actions, respectively. It can be seen that the backup actions are able to improve the completion rate significantly in all five case studies. With a higher completion rate, the total disassembly time increases as well.

The average disassembly time per element calculated by $\bar{t} = \frac{T}{P \cdot m}$ is shown in Fig. 10(b), (d), (f), (h) and (j). Using backup actions does not increase the average disassembly time per element significantly. The backup actions can largely increase the completion rate of a disassembly process in all five cases and reduce the time variance especially for the 'Product B', the worm gear and the gear pump. Without backup actions, the solutions are closely distributed in a small range with much lower completion rate and slightly lower disassembly time.

The range of solutions with and without backup are summarized in Table IV. The completion rate in all case studies triples due to the backup actions. It proves that the backup actions can reduce disassembly time per task, as well as increase the reliability of a plan and avoid frequent online re-planning.

D. The Efficiency of the DS-MOEA on Robotic Disassembly Sequence Planning

The HV and IGD results of different MOEAs on solving the robotic disassembly sequence planning problems with task

TABLE V
THE HV AND IGD RESULTS OF THE FIVE MOEAS IN THE THREE CASE STUDIES

Product	Metric	IBEA	MOEA/D	NSGA-II	NSGA-III	DS-MOEA
'Product C'	HV	0.8920	0.7943	0.8106	0.5278	0.8926
	IGD	0.6155	2.2956	2.0087	3.2147	0.6682
'Product B'	HV	0.8976	0.7420	0.7745	0.6252	0.9019
	IGD	0.5246	2.0639	1.9043	2.0235	0.5539
Worm gear	HV	0.8215	0.6242	0.6567	0.8085	0.8608
	IGD	0.6460	1.8196	1.6959	0.7211	0.5889
Motor reducer	HV	0.8752	0.6721	0.6472	0.7647	0.9042
	IGD	0.6056	3.2484	3.2394	1.1434	0.5778
gear pump	HV	0.8361	0.6296	0.6539	0.7567	0.8610
	IGD	0.7347	4.2432	3.8756	1.7465	0.7065

TABLE VI
THE ϵ -INDICATOR MATRIX OF THE FIVE MOEAS ON THE THREE CASE STUDIES

'Product C'	ϵ -indicator	IBEA	MOEA/D	NSGA-II	NSGA-III	DS-MOEA	winning number
	IBEA	0	0.1095	0.1003	0.0199	0.1527	3
	MOEA/D	0.2161	0	0.1136	0.0302	0.2175	1
	NSGA-II	0.1800	0.0891	0	0.0172	0.1952	2
	NSGA-III	0.4292	0.3798	0.3861	0	0.4437	0
	DS-MOEA	0.1504	0.0762	0.0719	0.0141	0	4
'Product B'	ϵ -indicator	IBEA	MOEA/D	NSGA-II	NSGA-III	DS-MOEA	winning number
	IBEA	0	0.0454	0.0753	0.0354	0.1252	3
	MOEA/D	0.2424	0	0.1238	0.0686	0.2672	1
	NSGA-II	0.2465	0.0811	0	0.0535	0.2178	2
	NSGA-III	0.3662	0.2146	0.2482	0	0.3510	0
	DS-MOEA	0.1214	0.0679	0.0693	0.0294	0	4
Worm gear	ϵ -indicator	IBEA	MOEA/D	NSGA-II	NSGA-III	DS-MOEA	winning number
	IBEA	0	0.0565	0.0624	0.1106	0.1745	3
	MOEA/D	0.3715	0	0.0936	0.2626	0.3689	0
	NSGA-II	0.3409	0.0601	0	0.2377	0.3453	1
	NSGA-III	0.2060	0.0540	0.0564	0	0.2325	2
	DS-MOEA	0.1525	0.0418	0.0469	0.1064	0	4
Motor reducer	ϵ -indicator	IBEA	MOEA/D	NSGA-II	NSGA-III	DS-MOEA	winning number
	IBEA	0	0.0265	0.0149	0.0650	0.1519	3
	MOEA/D	0.3309	0	0.0656	0.2532	0.3224	1
	NSGA-II	0.3425	0.0985	0	0.2498	0.3498	0
	NSGA-III	0.2049	0.0761	0.0542	0	0.2372	2
	DS-MOEA	0.1365	0.0363	0.0304	0.0827	0	4
Gear pump	ϵ -indicator	IBEA	MOEA/D	NSGA-II	NSGA-III	DS-MOEA	winning number
	IBEA	0	0.0397	0.0539	0.1369	0.1771	3
	MOEA/D	0.3485	0	0.1180	0.3083	0.3585	0
	NSGA-II	0.3269	0.0688	0	0.2809	0.3356	1
	NSGA-III	0.1556	0.0392	0.0393	0	0.1716	2
	DS-MOEA	0.1274	0.0354	0.0308	0.1125	0	4

failures are listed in Table V. In the small scale cases, i.e., for the 'Product C' and the 'Product B', The HV value of the DS-MOEA is better than that of the IBEA. However, the IGD value of the DS-MOEA is slightly worse than that of the IBEA. As the number of elements to be disassembled increases, the DS-MOEA performs remarkably better in the rest three case studies, with high HVs and low IGDs.

In addition, the additive ϵ -indicator is introduced to make a pair comparison between two of the five MOEAs. Table VI shows the the average value of the indicator for each pair in 50 independent runs. The first row of Table VI demonstrates $\epsilon_+(U_{IBEA}, U_*)$, where $*$ can be the rest of the four algorithms; while the second, the third, the fourth and the fifth rows represent $\epsilon_+(U_{MOEA/D}, U_*)$; $\epsilon_+(U_{NSGA-II}, U_*)$; $\epsilon_+(U_{NSGA-III}, U_*)$; and $\epsilon_+(U_{DS-MOEA}, U_*)$, respectively. From the winning number shown in the last column of Table VI it can be seen that the quality rank of the five algorithms for the

first two small-scale cases is DS-MOEA>IBEA>NSGA-II>MOEA/D>NSGA-III. When the number of elements to be disassembled increases, the quality rank for the rest three cases becomes DS-MOEA>IBEA>NSGA-III>MOEA/D $\simeq$ NSGA-II.

In summary, the experimental results show that NSGA-III is more suitable for the large-scale decision making, while NSGA-II is more useful for the small-scale scenarios. The crowd distance assignment seems to be more appropriate for the small-scale disassembly. But since the ϵ -indicator is adopted as a supplement in the proposed algorithm, more dynamic candidates should be filtered from the top fronts. Thus, the selection strategy of NSGA-III is more suitable to be used in combination with the ϵ -indicator than that of NSGA-II. The results have also proved the performance of the DS-MOEA is high and stable for both the small-scale disassembly and the large-scale disassembly compared with the original IBEA and NSGA-III. **It is able to find better disassembly plans**

with lower expected disassembly time and higher success rate.

VI. CONCLUSION AND FUTURE WORKS

Uncertainties in EOL conditions make disassembly difficult to robotize. A key problem is that existing industrial automation techniques use a pre-determined approach and cannot deal with unforeseen failed disassembly operations. This research modeled the completion rate of a disassembly plan and introduces backup actions to allow robotic systems to recover from failures automatically, and thus improved the robustness of robotic disassembly. A practical way to calculate the expected disassembly time to disassembly each component or subassembly and the completion rate of a disassembly plan was proposed. The robotic disassembly sequence planning problem with task failure was presented mathematically as a bi-objective optimization problem. A dual-selection multi-objective evolutionary algorithm was designed for the proposed model. Experimental results proved that using automatic backup actions is able to increase the completion rate of a plan by three times within a reasonable disassembly time. The proposed method also outperformed other typical evolutionary approaches.

In practice, more complex backup actions with nonlinear motions are required. In future works, our research will focus on the implementation of these backup actions in practice and the extension of these backup actions to different scenarios. The encoding scheme of the backup actions designed in this paper and the corresponding PPX operator will be modified accordingly. To further increase the success rate of an automatic disassembly process, it is a great necessity to introduce online re-planning module with dynamic disassembly decision-making to response the cases when all backup actions failed.

ACKNOWLEDGMENT

This work is supported by the National Key Research and Development Program of China (Grant No. 2018YFB1701601), and in part by the Royal Society and the Natural Science Foundation of China (IEC\NSFC\181018) and the Engineering and Physical Sciences Research Council (EPSRC Grant No. EP/N018524/1).

REFERENCES

- [1] S. Vongbunyong, S. Kara, and M. Pagnucco. Learning and revision in cognitive robotics disassembly automation. *Robotics and Computer-Integrated Manufacturing*, 34:79–94, 2015.
- [2] G. Tian, M. C. Zhou, and P. Li. Disassembly sequence planning considering fuzzy component quality and varying operational cost. *IEEE Transactions on Automation Science and Engineering*, 15(2):748–760, 2017.
- [3] S. Ghandi and E. Masehian. Review and taxonomies of assembly and disassembly path planning problems and approaches. *Computer-Aided Design*, 67:58–86, 2015.
- [4] N. M. DiFilippo and M. K. Jouaneh. A system combining force and vision sensing for automated screw removal on laptops. *IEEE Transactions on Automation Science and Engineering*, 15(2):887–895, 2017.
- [5] G. Tian, M. Zhou, J. Chu, and Y. Liu. Probability evaluation models of product disassembly cost subject to random removal time and different removal labor cost. *International Journal of Production Research*, 9(2):288–295, 2012.
- [6] S. A. Reveliotis. Uncertainty management in optimal disassembly planning through learning-based strategies. *IEE Transactions*, 39(6):645–658, 2007.
- [7] H. Cheng, W. Xu, Q. Ai, Q. Liu, Z. Zhou, and D. T. Pham. Manufacturing capability assessment for human-robot collaborative disassembly based on multi-data fusion. *Procedia Manufacturing*, 10:26–36, 2017.
- [8] F. T. Altekın, Z. P. Bayındır, and V. Gümüşkaya. Remedial actions for disassembly lines with stochastic task times. *Computers & Industrial Engineering*, 99:78–96, 2016.
- [9] K. Deb and H. Jain. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints. *IEEE Transactions on Evolutionary Computation*, 18(4):577–601, 2014.
- [10] E. Zitzler and S. Künzli. Indicator-based selection in multiobjective search. In *Proc. Springer International Conference on Parallel Problem Solving from Nature (PPSN)*, pages 832–842, Birmingham, UK, September 2004.
- [11] E. Kongar and S. M. Gupta. Disassembly sequencing using genetic algorithm. *International Journal of Advanced Manufacturing Technology*, 30:497–506, 2006.
- [12] Y. M. Huang and C. T. Huang. Disassembly matrix for disassembly processes of products. *International Journal of Production Research*, 40(2):255–273, 2002.
- [13] H. Wang. Disassembly sequence planning for end-of-life products. Master's thesis, University of Manitoba, Winnipeg, 2016.
- [14] Y. Feng, Y. Gao, G. Tian, Z. Li, H. Hu, and H. Zheng. Flexible process planning and end-of-life decision-making for product recovery optimization based on hybrid disassembly. *IEEE Transactions on Automation Science and Engineering*, 16(1):311–326, 2018.
- [15] J. Liu, Z. Zhou, D. T. Pham, W. Xu, C. Ji, and Q. Liu. Robotic disassembly sequence planning using enhanced discrete bees algorithm in remanufacturing. *International Journal of Production Research*, 56(9):3134–3151, 2018.
- [16] B. Adenso-Díaz, S. García-Carbalal, and S. M. Gupta. A path-relinking approach for a bi-criteria disassembly sequencing problem. *Computers & Operations Research*, 35(12):3989–3997, 2008.
- [17] S. S. Smith and W. H. Chen. Rule-based recursive selective disassembly sequence planning for green design. *Advanced Engineering Informatics*, 25(1):77–87, 2011.
- [18] E. Kongar and S. M. Gupta. Disassembly sequencing using genetic algorithm. *The International Journal of Advanced Manufacturing Technology*, 30(5-6):497–506, 2006.
- [19] T. F. Go, D. A. Wahab, M. N. Ab. Rahman, R. Ramli, and C. H. Azhari. Disassemblability of end-of-life vehicle: a critical review of evaluation methods. *Journal of Cleaner Production*, 19(13):1536–1546, 2011.
- [20] X. Guo, S. Liu, M. C. Zhou, and G. Tian. Disassembly sequence optimization for large-scale products with multiresource constraints using scatter search and petri nets. *IEEE Transactions on Cybernetics*, 46(11):2435–2446, 2016.
- [21] F. Tao, L. Bi, Y. Zuo, and A. Y. C. Nee. Partial/parallel disassembly sequence planning for complex products. *Journal of Manufacturing Science and Engineering*, 140(1):011016, 2018.
- [22] K. Meng, P. Lou, X. Peng, and V. Prybutok. An improved co-evolutionary algorithm for green manufacturing by integration of recovery option selection and disassembly planning for end-of-life products. *International Journal of Production Research*, 54(18):5567–5593, 2016.
- [23] X. Guo, M. Zhou, S. Liu, and L. Qi. Multiresource-constrained selective disassembly with maximal profit and minimal energy consumption. *IEEE Transactions on Automation Science and Engineering*, 2020.
- [24] A. J. D. Lambert. Disassembly sequencing: a survey. *International Journal of Production Research*, 41:3721–3759, 2003.
- [25] Y. Ren, L. Meng, C. Zhang, F. Zhao, U. Saif, A. Huang, G. P. Mendis, and J. W. Sutherland. An efficient metaheuristics for a sequence-dependent disassembly planning. *Journal of Cleaner Production*, 245:118644, 2020.
- [26] S. Ghandi and E. Masehian. A breakout local search (bls) method for solving the assembly sequence planning problem. *Engineering Applications of Artificial Intelligence*, 39:245–266, 2015.
- [27] X. Guo, S. Liu, M. C. Zhou, and G. Tian. Dual-objective program and scatter search for the optimization of disassembly sequences subject to multiresource constraints. *IEEE Transactions on Automation Science and Engineering*, 15(3):1091–1103, 2018.
- [28] Y. Ren, C. Zhang, F. Zhao, H. Xiao, and G. Tian. An asynchronous parallel disassembly planning based on genetic algorithm. *European Journal of Operational Research*, 269(2):647–660, 2018.

- [29] A. ElSayed, E. Kongar, S. M. Gupta, and T. Sobh. A robotic-driven disassembly sequence generator for end-of-life electronic products. *Journal of Intelligent & Robotic Systems*, 68(1):43–52, 2012.
- [30] G. Tian, M. Zhou, and J. Chu. A chance constrained programming approach to determine the optimal disassembly sequence. *IEEE Transactions on Automation Science and Engineering*, 10(4):1004–1013, 2013.
- [31] Y. Laili, F. Tao, D. T. Pham, Y. Wang, and L. Zhang. Robotic disassembly re-planning using a two-pointer detection strategy and a super-fast bees algorithm. *Robotics and Computer-Integrated Manufacturing*, 59:130–142, 2019.
- [32] E. Kroll and B. S. Carver. Disassembly analysis through time estimation and other metrics. *Robotics and Computer Integral Manufacturing*, 15(3):191–200, 1999.
- [33] M. Marconi, M. Germani, M. Mandolini, and C. Favi. Applying data mining technique to disassembly sequence planning: a method to assess effective disassembly time of industrial products. *International Journal of Production Research*, 57(2):599–623, 2019.
- [34] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.
- [35] H. E. Tseng, C. C. Chang, S. C. Lee, and Y. M. Huang. Block-based genetic algorithm for disassembly sequence planning. *Expert Systems with Applications*, 96(15):492–505, 2018.
- [36] GrabCAD. <http://www.michaelsshell.org/tex/ieeetran/>.
- [37] K. Li, R. Wang, T. Zhang, and H. Ishibuchi. Evolutionary many-objective optimization: a comparative study of the state-of-the-art. *IEEE Access*, 10.1109/ACCESS.2018.2832181, pages 26194 – 26214, 2018.
- [38] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. Da Fonseca. Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Transactions on Evolutionary Computation*, 7(2):117–132, 2003.
- [39] M. Li and X. Yao. Quality evaluation of solution sets in multiobjective optimisation: a survey. *ACM Computing Surveys*, 52(2):26, 2019.

Yuanjun Laili Yuanjun Laili received the BS, MS, and Ph.D Degree from the School of Automation Science and Electrical Engineering at Beihang University. She is an Assistant Professor of the School of Automation Science and Electrical Engineering, Beihang University. She is a member of IEEE Robotics and Automation Society and SCS (The Society For Modeling and Simulation International). She is also an Associate Editor of “International Journal of Modeling, Simulation, and Scientific Computing”.

She has won the “Young Talent Lift Project” supported by China Association for Science and Technology and the “Young Simulation Scientist Award” from SCS. Her main research interests are in the areas of intelligent optimization, modeling and simulation of manufacturing systems.

Xiang Li Xiang Li received the BS Degree from the School of Automation Science and Electrical Engineering at Beihang University. Now she is studying as a postgraduate in the School of Automation Science and Electrical Engineering at Beihang University. Her research interests include evolutionary optimization for large-scale cloud computing system and robotic disassembly for remanufacturing.

Yongjing Wang Yongjing wang is a research fellow within the Autonomous Remanufacturing Laboratory (ARL) at the University of Birmingham. He has been leading the design and developments of several robotic disassembly systems at Birmingham. His research has been reported in over 13 international media. Yongjing is a Fellow of the Higher Education Academy, a member of the Institute of Technology and a winner of Royal Society International Exchange Award in 2019.

Lei Ren Lei Ren is an associate professor and the deputy head of Cloud Manufacturing Research Center at School of Automation Science and Electrical Engineering in Beihang University, and a senior research scientist at Engineering Researching Center of Complex Product Advanced Manufacturing Systems, Ministry of Education, China. He received a Ph.D. degree in computer science from the Institute of Software, Chinese Academy of Sciences in 2009. His research interests include big data analytics and applications. He has published 50+ papers and got

2000+ citations according to Google Scholar. He edited a book entitled “Challenges and Opportunity with Big Data” published by Springer LNCS. He served as an associate editor of SIMULATION: Transactions of the Society for Modeling and Simulation International, and reviewers for journals such as IEEE TII, TSMC, ACM TCC. He also served as a TPC co-chair for the 19th Monterey Workshop on Big Data, 5th International Conference on Enterprise Systems, etc. He is a member of IEEE, ACM, ASME and SCS.

Xiaokang Wang Xiaokang Wang received the Ph.D degree in Computer System Architecture from the Huazhong University of Science and Technology, Wuhan, China. Currently, he is a post-doctoral research fellow with the Department of Computer Science, St. Francis Xavier University, Antigonish, NS, Canada. His research interests include Cyber-Physical-Social Systems (CPSS), Big Data, Parallel and Distributed Computing, and Tensor Computing. He is a member of the IEEE and CCF.