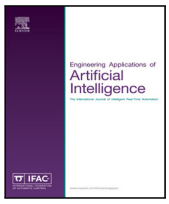
Engineering Applications of Artificial Intelligence 150 (2025) 110530

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai



Surrogate-assisted motion planning and layout design of robotic cellular manufacturing systems

Tomoya Kawabe *, Tatsushi Nishi, Ziang Liu, Tomofumi Fujiwara

Graduate School of Environmental, Life and Natural Science and Technology, Okayama University, 3-1-1 Tsushima-naka, Kita-ku, Okayama City, Okayama 700-8530, Japan

ARTICLE INFO

Keywords:
Robotics
Cellular manufacturing
Layout design
Sequence-pair
Motion planning
Surrogate optimization
Machine learning
Artificial intelligence

ABSTRACT

A surrogate-assisted multi-objective evolutionary algorithm is proposed for simultaneous optimization of robot motion planning and layout design in robotic cellular manufacturing systems. A sequence-pair is used to represent the layout of components in a robotic cell to avoid overlapping in the evolutionary computation. The robot motion planning with Rapidly exploring Random Trees Star (RRT*) is applied to compute the total operation time of a robot arm for each layout. Non-dominated Sorting Genetic Algorithm II (NSGA-II) is used to minimize the total required layout area and the operation time for a robot arm. The proposed surrogate model can estimate the robot's operation time with 98% of accuracy without explicit computations of the motion planning algorithm. The experimental results with a physical 6 Degree of Freedom (DOF) manipulator show that the total computation time is approximately 1/400, significantly shorter than the conventional methods.

1. Introduction

Cellular manufacturing has several advantages, including improved system efficiency and quality of products, reduction in setup time, work-in-process inventory, and material flow (Salimpour et al., 2021). Robotic cellular manufacturing systems (RCMSs) are a new type of manufacturing system in which one or more robots and accessory equipment are involved to conduct several assembly operations that had been performed by human operators in conventional cellular manufacturing systems (Izui et al., 2013).

The optimization of layout design and motion planning for robots has great economic significance for RCMSs, such as in reducing total costs and necessary areas, maximizing throughput, and enhancing efficiency to exploit the advantages fully. Traditional layout planning methods highly rely on the planner's knowledge and the layout design of robotic cells and motion planning for robot arms are often separated. In the layout design stage, the positions of robots and workstations are determined in a robotic cell. Then, the motion planning is conducted on the condition that the layout design is fixed. In those cases, the dynamic performance of robotic motion planning is sometimes neglected at the layout design stage. As a result, it is difficult to find an optimal layout design and optimal motion planning considering the manipulability of the robot arms reducing total cycle time within a minimum necessary area. Therefore, it is important to develop an efficient method to find the optimal layout design method considering motion planning for the RCMSs. However, the computational burden of optimizing both layout design and motion planning is extremely high because the layout design problem without motion planning is Nondeterministic Polynomial (NP) hard. Simulation-based optimization methods sometimes require more than two or three days to derive near-optimal solutions. In such cases, it is difficult to obtain the design and motion planning under different configurations. Once the approximation of robot operating time is used, the layout cannot be executed in real-world environments. The acceleration of computation time for layout design and motion planning without loss of precision is highly required. It motivates the surrogate-assisted motion planning and layout design of RCMSs proposed in this paper.

Simulation-based optimization approaches using Genetic Algorithms (GA) have been utilized to solve layout optimization problems (Leiber et al., 2022). In most cases, the evolutionary computation of motion planning for each layout design requires extensive computational efforts. The sequence-pair proposed by Murata et al. (1996) for Very-Large-Scale Integration (VLSI) layout design has been effectively used to represent the geometric relationships among modules for layout design problems. A major advantage is that it can eliminate the overlapping between modules. Several previous studies have discussed the applications of the sequence-pair for RCMSs. Izui et al. (2013) proposed a multi-objective layout optimization method for the RCMSs using

* Corresponding author.
E-mail address: nishi.tatsushi@okayama-u.ac.jp (T. Nishi).

https://doi.org/10.1016/j.engappai.2025.110530
Received 20 August 2024; Received in revised form 21 January 2025; Accepted 5 March 2025
Available online 25 March 2025

sequence-pair representation. The objectives are to minimize the layout area and the operation time while maximizing the manipulability. Along the same lines, Lim et al. (2017) have proposed multi-objective hybrid algorithms for layout optimization problems. These previous studies have successfully obtained the Pareto set of optimal solutions. No previous studies have reported the simultaneous optimization of robotic cell layout and precise motion planning at the same time. Generally, the simultaneous optimization of layout design and motion planning generally requires extensive computational efforts. A surrogate-assisted optimization is required to realize the simultaneous computation within a realistic computation time for the simultaneous optimization.

This paper presents a surrogate-assisted simultaneous motion planning and layout optimization method for RCMSs. The sequence-pair is used to represent the layout optimization of the robotic cell. For a given sequence-pair, motion planning is conducted to compute the total operation time. To reduce the computational effort of the motion planning algorithm, machine learning models are employed to estimate a precise operation time to complete the task. Since a certain layout design candidate may not be a feasible solution, the feasibility of a sequence-pair is also evaluated in the motion planning process. Rapidly Exploring Random Trees (RRT) are used to search for the optimal motion. To solve the multi-objective optimization problem, a surrogate-assisted Non-dominated Sorting Genetic Algorithm II (NSGA-II) is proposed to derive the Pareto optimal solution. The main contributions of the paper are as follows.

- The multi-objective optimization of simultaneous motion planning and layout design is achieved for RCMSs to minimize the area and the total operation time. Detailed collision-free motion planning for a single robot arm is incorporated into the design of robot cells.
- Surrogate-assisted multi-objective optimization method can significantly reduce the computation time, which previously took several hours.
- The proposed method is applied to a 6 Degree of Freedom (DOF) robot manipulator. The experimental results suggest the effectiveness of the proposed algorithm.

This study focuses on the simultaneous optimization of layout design and motion planning for RCMSs. Research into Layout design of cellular manufacturing systems (CMSs) has a long history. Examples of previous studies include several works (Akturk and Turkcan, 2000; Kia et al., 2012; Bayram and Şahin, 2016; Feng et al., 2018; Forghani and Fatemi Ghomi, 2020). With the development of industrial robots, the interest in efficient layout design and motion planning for RCMSs has increased. The RCMS design problem determines the positions of robots and workstations in a robotic cell. Our research group has extended this problem to various applications, such as the design of pick-and-place systems in production lines (Nonoyama et al., 2022), packaging systems in logistics (Mikyu et al., 2024), and task planning in service robots (Kawabe et al., 2024). Given the broad applicability of RCMS optimization, this study addresses the general problem of layout design and motion planning rather than focusing on a specific application. The proposed approach provides a method to efficiently solve the optimization problem of RCMSs within a limited computation time, which is a significant contribution to the field of manufacturing systems and expected to be widely applicable to various industries.

The remaining part of this paper is organized into the following sections. Section 2 describes the literature review. Section 3 introduces integrated motion planning and layout design problem for RCMSs. Section 4 presents the surrogate-assisted evolutionary optimization algorithm for simultaneous optimization using accurate robot operation times. The experimental results using a physical robot manipulator are shown in Section 5. Section 6 summarizes the conclusions and indicates future research directions.

2. Literature review

Related references are classified into cellular manufacturing system optimization, RCIMs, sequence pair representation, and surrogate-assisted optimization.

2.1. Cellular manufacturing system optimization

Akturk and Turkcan (2000) have proposed a local search heuristic to simultaneously optimize the cell formation problem and the layout problem. Kia et al. (2012) have proposed a mixed integer non-linear programming (MINLP) model for simultaneously optimizing cell formation, inter-cell and intra-cell layout problem with dynamic CMS. Kia et al. (2012) and Bayram and Şahin (2016) have proposed two hybrid meta-heuristic algorithms for the dynamic CMS design problem showing that their approaches have better performance in terms of computation time and solution quality. Feng et al. (2018) have also focused on the integrated cell formation, inter- and intra-cell layout problem. Since the two-dimensional coordinate is used to represent the locations of machines, the components in cells can have different shapes. Forghani and Fatemi Ghomi (2020) have introduced the virtual CMS to the integrated CMS optimization problem.

Most of these previous studies assume that the processing times of parts for machines are known, and the single or multi-row layout of equal area facilities is considered. However, facilities may have different shapes in CMSs. In traditional CMSs, families of parts are produced by a single line or cell of machines operated by machinists (Helms, 2021). Recent developments in the field of industrial robots have led to a growing interest in RCMSs. In an RCMS, since the optimal robot motions change with the intra-cell layout, it is unrealistic to assume that the processing time is constant. Therefore, it is necessary to simultaneously solve the motion planning and layout design problem in RCMSs.

2.2. Robotic cellular manufacturing system optimization

Nonoyama et al. (2022) have focused on the optimal motion planning and layout design problem. Since their study only involves a robot, a conveyor belt, and workpieces, the non-overlapping constraint can be easily implemented by restricting the range of the robot's feasible locations. However, since CMSs may involve multiple robots and workstations, it is difficult to handle the non-overlapping constraints when using the coordinate representation. Shiller (1989) developed an application for the design of work cells and the optimization of robot operations. However, their formulation considers the pre-defined task of planning robot movements through multiple waypoints, with the focus primarily on motion planning. The work cell design focuses solely on arranging obstacles in a way that prevents interference with the robot, and it is not sufficient as a simultaneous optimization method for both placement and motion planning.

2.3. Sequence-pair representation

The sequence-pair proposed initially by Murata et al. (1996) for VLSI layout design, can be used to effectively represent the geometric relationships among modules for layout designs. A major advantage is that it can eliminate the overlapping between modules. Another advantage is that sequence-pair can also handle the unequal-area facility layout problem. It has been applied in various fields, such as facility layout problems (Liu and Meller, 2007; Meller et al., 2007), berth allocation problems (Mohammadi and Forghani, 2019) and scheduling problems (Kozik, 2017). During the last decade, several researchers have applied sequence-pair representation in the design of CMSs. Izui et al. (2013) have proposed a multi-objective optimization model with three important decision criteria for robot CMSs design, including

Table 1
Related works on cellular manufacturing system optimization.

Reference	CF	Inter-cell layout	Intra-cell layout	Facilities	Operation time	Objective	Motion planning	Real robot	Algorithm
Kia et al. (2012)	✓	✓	✓	Equal	Constant	Single	–	–	Heuristic
Akturk and Turkcan (2000)	✓	✓	✓	Equal	Constant	Single	–	–	SA
Bayram and Şahin (2016)	✓	✓	✓	Equal	Constant	Single	–	–	HMLP[a]
Feng et al. (2018)	✓	✓	✓	Unequal	Constant	Single	–	–	HMLP
Forghani et al. (2020)	✓	✓	✓	Unequal	Constant	Single	–	–	SA, GA, MA
Izui et al. (2013)	–	–	✓	Unequal	Estimation	Multiple	–	–	NSGA-II
Lim et al. (2016)	–	–	✓	Unequal	Estimation	Multiple	–	–	HM[b]
Suemitsu et al. (2016)	–	–	✓	Unequal	Estimation	Multiple	–	–	NSGA-II
Lim et al. (2017)	–	–	✓	Unequal	Estimation	Multiple	–	–	HM
Kawabe et al. (2022)	–	–	✓	Unequal	Accurate	Multiple	✓	✓	NSGA-II
Current study	–	–	✓	Unequal	Data-driven	Multiple	✓	✓	NSGA-II

[a] Hybrid Metaheuristics with Linear Programming.
[b] Hybrid Metaheuristics.

operation time, feasibility of assembly tasks, and layout area. A Non-dominated sorting genetic algorithm, NSGA-II (Deb et al., 2002) has been used to solve this problem. Lim et al. (2016) have examined the performance of sequence-pair and B*-trees (Chang et al., 2000) on the CMS design problem of Izui et al. (2013). Later, Lim et al. (2017) have proposed four hybrid algorithms and compared their performance on the RCMS design problem. On the basis of Izui et al.'s model, (Izui et al., 2013; Suemitsu et al., 2016) have integrated the task scheduling problem to the layout design of RCMSs. Forghani et al. (2020) have proposed two integrated Simulated Annealing (SA) algorithms for an integrated cell formation and layout design problem. Recently, Kawabe et al. (2022) have proposed an optimization method for simultaneous motion planning and intra-cell layout design in RCMSs. Overall, these previous studies (Izui et al., 2013; Lim et al., 2016, 2017; Suemitsu et al., 2016) suggest the effectiveness of the sequence-pair representation in RCMS design problems. All these studies assume that the operation time for a certain layout can be estimated by the maximum joint motion time, which is not the exact operation time for a layout in RCMSs. Kawabe et al. (2022) have successfully developed an integrated optimization algorithm for motion planning and layout design to obtain the exact operation time. However, the computation time requires more than 30.6 h since all the candidate solutions are enumerated in their algorithm. We summarize the features of the current study and the related works on cellular manufacturing system optimization in Table 1.

2.4. Surrogate-assisted optimization

Because of the complexity of the CMS design, metaheuristic algorithms are commonly used for the optimization of CMSs as shown in Table 1. However, to obtain a satisfactory solution for a complex optimization problem, a large number of computations of the function evaluations are commonly required for metaheuristic algorithms (Tong et al., 2021). In many real-world problems, the function evaluation is the most time-consuming component of evolutionary algorithms (Coello Coello et al., 2007). Furthermore, multi-objective evolutionary algorithms, such as NSGA-II, may need more execution time for better performance compared with single-objective evolutionary algorithms (Coello Coello et al., 2007).

In recent years, researchers have shown an increased interest in integrating machine learning into metaheuristic algorithms (Karimi-Mamaghan et al., 2022). Supervised learning, unsupervised learning, and reinforcement learning are the three main types of machine learning algorithms. Supervised learning can be used to predict/estimate the output of a function given input data using a labeled dataset. Supervised learning has been used in various fields, such as the remaining useful life prediction (Zhang et al., 2023a,b). An important application of machine learning in metaheuristic algorithms is a functional approximation, that has been known as surrogate-assisted evolutionary algorithms (Jin, 2011). For optimization problems with expensive objective functions, using a surrogate model may significantly reduce time, space, and computing costs (Bartz-Beielstein and Zaefferer, 2017). Also, there has been a growing number of publications focusing on the real-world applications of surrogate-assisted evolutionary algorithms, such as regional trauma system design (Wang and Jin, 2021), wind farm layout optimization (Long and Li, 2020; Li et al., 2022), blast furnace optimization (Chugh et al., 2017), scheduling problems (Togo et al., 2022), neural architecture search (Calisto and Lai-Yuen, 2021), and supply chain management (Liu and Nishi, 2024).

To date, very few studies have investigated the application of surrogate-assisted evolutionary algorithms in industrial robots. Zhang and Yan (2021) have developed a surrogate model by using an artificial neural network to reveal the relations between operation parameters and the energy consumption of an industrial robot. Genetic algorithms (GA) have been used to optimize the parameters to minimize energy consumption based on the surrogate model. A recent study by Liu et al. (2024) has proposed a surrogate-assisted optimization method for packing problems in the logistics industry. The surrogate model has been used to accelerate the computation of the objective function in the evolutionary algorithm. The results suggest that the proposed method can significantly reduce the computation time by more than 90% compared to the non-surrogate method in Mikyu et al. (2024). To the best of our knowledge, this is the first study to implement surrogate-assisted evolutionary optimization to the RCMSs design problems.

3. Integrated motion planning and layout design problem for robotic cellular manufacturing systems

3.1. Problem description

This study aims to address the motion planning and intra-cell layout design problem of RCMSs in a static environment. We focus on the optimization problem for a robot cell in a two-dimensional space that consists of a single robot, an assembly table, and multiple parts boxes. The robot is responsible for moving parts from the boxes to the assembly table in a predetermined sequence and carrying out the assembly operations on the assembly table. This study assumes a static manufacturing environment, where the sequence of assembly tasks is given in advance.

Let $I = \{0, 1, \ldots, n\}$ be a set of components in a robotic cell which includes a robot, an assembly table and parts boxes. The robot is represented by 0, the assembly table is represented by 1, and the parts boxes are represented by $2, 3, \ldots, n$. Each component $i \in I$ has two modes, the initial mode and the 90-degree rotated mode which are represented by $k \in K$, where $K = \{ini, rot\}$. Each mode k of component i specifies its width w_i^k and length l_i^k. For example, w_3^{rot} represents the width of component 3 in its 90-degree rotation mode. Let A represent the area of an intra-cell layout of a robotic cell, and T represent the total operation time required for a robot to complete the assembly task

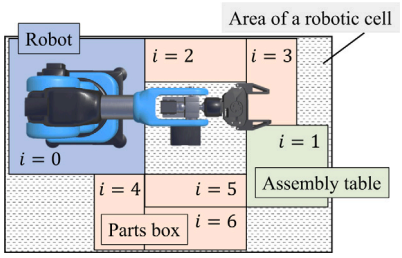


Fig. 1. Area of a robotic cell.

which is given as an operation sequence. The objectives are to minimize the area A and to minimize the total operation time T. Similar to the previous studies (Izui et al., 2013; Lim et al., 2016, 2017; Suemitsu et al., 2016), we define A as the minimum area of a rectangle that can encompass all the components in a robotic cell, as illustrated in Fig. 1. Here, it is clear that the order of the numbers labeled on each component does not affect the area of the layout and operation time of the robot.

Also, the parts boxes are labeled with a number that denotes their assembly sequence as shown in Fig. 1. To perform the assembly task, the robot picks up and transports parts from the box with the lowest numbered label to the assembly table first, and then moves on to the box with the second-smallest number, and so on, until the parts in the box with the highest numbered label are transported to the table.

The total operation time T consists of the pick-up time, transport time, and the assembly time. The cell layout only has a major impact on the transport time, it is worth noting that our model only considers the transport time in a robotic cell for an assembly task and the time for the assembly and pick-up operation is sufficiently small compared with the transport time for simplicity. We assume that the assembly sequence is arbitrarily given because the total operation time does not depend on the assembly sequence. Multiple assembly tasks cannot be performed during the same time period. All tasks must be performed as soon as possible after a given task is completed.

Conventional studies did not consider the motion trajectory of the robot in their layout design problem. However, the determination of layout design in RCMSs is highly related to the trajectory of motion planning for the robot. For example, the total operation time may change depending on the posture and path of the robot and the positioning of the robot and boxes.

Overall, we state the joint motion planning and intra-cell layout design optimization problem as follows.

Given parameters

Number and dimensions of boxes, dimensions of the assembly table and robot, assembly sequence, number of links in the robot, the length of each link of the robot, and physical parameters to describe the dynamics of the robot: range of movement of each link, the angular velocity of each link, the joint angular acceleration of each link, the mass of each joint, coordinates of the center of gravity of each joint, moment of inertia of each joint, and gravitational acceleration applied to the robot.

Decision variables

- Two-dimensional layout of the robot, assembly table, and parts boxes
- Motion trajectory of the robot

Objective function In the layout optimization, the total required area and the total operation time are minimized because they are related to the costs. We use the following two objective functions.

- To minimize the total area A of a intra-cell layout
- To minimize the total operation time T of the robot for a given assembly sequence

The operation time T has been approximated by the following equation in Izui et al. (2013) and Suemitsu et al. (2016) without generating a motion trajectory of the robot.

$$T = \sum_{i=1}^{h} \max_{k} \left[\frac{|\bar{q_k}^i|}{v_k} \right] \tag{1}$$

where k is the number of each joint of the robot, v_k is the maximum angular velocity of joint kth, and $|\bar{q_k}^i|$ is the amount of angular change of joint kth during ith motion. There is an assumption that the angular velocity of the robot arm is constant in conventional studies. However, it does not hold in practice. In this problem definition, the detailed robot's operation time is computed when the joint angular velocity is changed. Determining the precise motion time is challenging because it involves solving nonlinear and complex dynamics equations when the initial position and target position of a robot are given. To compute the motion time, both inverse kinematics and motion planning are necessary.

Inverse kinematics

These are complex nonlinear equations that take into account elements such as mass, moment of inertia, and friction.

$$M(\boldsymbol{q})\ddot{\boldsymbol{q}} = -C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - G(\boldsymbol{q}) - J(\boldsymbol{q})^T F_{E_{xt}} + \boldsymbol{\tau} \tag{2}$$

where $M(\boldsymbol{q})$ is the mass matrix of each joint, $C(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is the Coriolis term, $G(\boldsymbol{q})$ is the gravity term, $J(\boldsymbol{q})$ is the Jacobian of each joint, $F_{E_{xt}}$ is the matrix of external forces on the robot, and $\boldsymbol{\tau}$ is the torque applied to each joint. Also, $\boldsymbol{q} = [q_1, q_2, \ldots]$ is the set of each joint angle, $\dot{\boldsymbol{q}}$ is each joint angular velocity vector, and $\ddot{\boldsymbol{q}}$ is each joint angular acceleration vector. The trajectory sequence $\zeta = \{(t_0, \boldsymbol{q}_0), (t_1, \boldsymbol{q}_1), \ldots, (t_K, \boldsymbol{q}_K)\}$ of the robot arm consisting of the set of each joint $\boldsymbol{q}$ at time t is derived by a motion planning algorithm. From the trajectory sequence $\zeta(t, \boldsymbol{q})$, the exact robot operation time T is determined from the last t_k in the trajectory sequence ζ. Solving the dynamic equations represented by Eq. (2) is required to determine the necessary torque and angular velocity for each joint.

Motion planning

In motion planning, the calculation of motion time takes into account the robot's angular velocity and acceleration. The motion planning algorithm is utilized to find a collision-free motion planning at each step of computing an objective function for each individual in the evolutionary computation. However, It is evident that the computational complexity would become exceedingly large when the precise robot motion times are determined for each individual in the evolutionary computation, Therefore, instead of pursuing precise analytical solutions, a surrogate model construction is required using deep learning techniques to accelerate computations.

Constraints

- The robot motions must be feasible without collisions.
- The parts boxes should not overlap with each other

3.2. Mathematical formulation

Let i be the index of the components in a robotic cell, and $I = \{0, 1, \ldots, n\}$ be the set of components in a robotic cell. Specifically, the robot is represented by 0, the assembly table is represented by 1, and the parts boxes are represented by $2, 3, \ldots, n$. Let π represent a layout in a robotic cell and the mode of components represented by $\mu(\pi) = (\mu_0(\pi), \mu_1(\pi), \ldots, \mu_n(\pi))$. For all $i \in I, \mu_i(\pi) \in K$. The location of component i is denoted by $(x_i(\pi), y_i(\pi))$ where the origins of the x, y coordinates of the layout is the position of the robot represented by

component 0. The length $l_{\max}(\pi)$ and width $w_{\max}(\pi)$ of a layout can be obtained by

$$l_{\max}(\pi) = \max_{\{i \in I\}}(x_i(\pi) + w_i^{(\mu_i(\pi))}) \tag{3}$$

$$w_{\max}(\pi) = \max_{\{i \in I\}}(y_i(\pi) + l_i^{(\mu_i(\pi))}) \tag{4}$$

The area of a robotic cell can be obtained by

$$f_{area}(\pi) = l_{\max}(\pi) \cdot w_{\max}(\pi) \tag{5}$$

Let a binary variable $p_{i,j}^d$ represent the relative position of component i and j in direction d, where $d \in \{left, right, behind, ahead\}$. For example, $p_{i,j}^{left} = 1$ means that component i is placed at the left of j. Therefore, the coordinates of components must satisfy the following inequalities:

$$p_{i,j}^{left} = 1 \Rightarrow x_i(\pi) + w_i^{(\mu_i(\pi))} \leq x_j(\pi) \tag{6}$$

$$p_{i,j}^{right} = 1 \Rightarrow x_j(\pi) + w_j^{(\mu_j(\pi))} \leq x_i(\pi) \tag{7}$$

$$p_{i,j}^{behind} = 1 \Rightarrow y_i(\pi) + l_i^{(\mu_i(\pi))} \leq y_j(\pi) \tag{8}$$

$$p_{i,j}^{ahead} = 1 \Rightarrow y_j(\pi) + l_j^{(\mu_j(\pi))} \leq y_i(\pi) \tag{9}$$

Let $u_{i,j}^d$ be a binary variable that represents the component i is placed at the left of j, and let $v_{i,j}^d$ be a binary variable that represents the component i is placed at the above of j. In addition, let W and L be the upper bound of the width and length of the layout, respectively. Constraints (6)–(9) can be rewritten as (10) and (11).

$$x_i(\pi) + w_i^{(\mu_i(\pi))} \leq x_j(\pi) + W(1 - u_{i,j}) \tag{10}$$

$$y_i(\pi) + l_i^{(\mu_i(\pi))} \leq y_j(\pi) + L(1 - v_{i,j}) \tag{11}$$

The operation time $f_{time}(\pi)$ is calculated by executing a motion planning. The constraint is that the robot arm does not collide with itself or with any object in its environment. Then, the integrated motion planning and layout design problem for RCMSs is formulated as follows.

$$\min \quad f_{area}(\pi),\ f_{time}(\pi) \tag{12}$$

$$\text{s.t.} \quad x_i(\pi) + w_i^{(\mu_i(\pi))} \leq x_j(\pi) + W(1 - u_{i,j}), \quad \forall i, j \in I, i \neq j \tag{13}$$

$$y_i(\pi) + l_i^{(\mu_i(\pi))} \leq y_j(\pi) + L(1 - v_{i,j}), \quad \forall i, j \in I, i \neq j \tag{14}$$

$$u_{i,j} + u_{j,i} + v_{i,j} + v_{j,i} \geq 1, \forall i, j \in I, i \neq j \tag{15}$$

$$x_i(\pi) + w_i^{(\mu_i(\pi))} \leq w_{\max}(\pi), \quad \forall i \in I \tag{16}$$

$$y_i(\pi) + l_i^{(\mu_i(\pi))} \leq l_{\max}(\pi), \quad \forall i \in I \tag{17}$$

$$x_i(\pi), y_i(\pi) \geq 0, \quad \forall i \in I \tag{18}$$

$$u_{i,j}, v_{i,j} \in \{0, 1\}, \quad \forall i, j \in I, i \neq j \tag{19}$$

Constraints (15) ensures that the at least one of the binary variables $u_{i,j}$, $u_{j,i}$, $v_{i,j}$, and $v_{j,i}$ is equal to 1. Constraints (16) and (17) are the constraints that the components are placed within the layout area. Constraints (18) are the non-negativity constraints for the coordinates of the components. Constraints (19) are the binary constraints for the relative positions of the components. An industrial application is a robotic cell layout optimization in small part assembly (SPA) (Zhang and Fang, 2013). Another practical application is the optimization of multifunctional cells in multi-robot systems (Nagele et al., 2020). The efficiency of multi-robot assembly operations is improved through placement optimization that takes into account the operation time of the robots.

4. Surrogate-assisted evolutionary algorithm

We introduce a surrogate-assisted evolutionary algorithm for solving the problem considering the exact robot operation time.

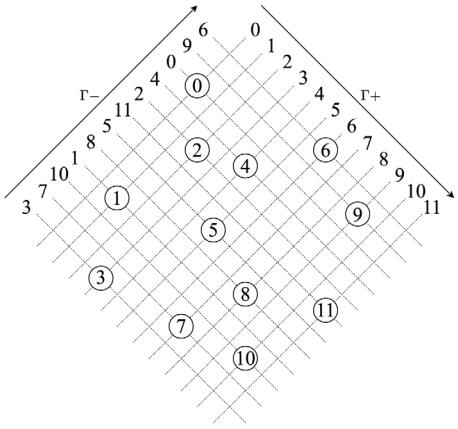


Fig. 2. An oblique grid for the sequence pair (Γ_+, Γ_-).

4.1. Sequence-pair representation

Sequence-pair representation consists of two related entity sequences that can be used to determine the relative positions of entities in a two-dimensional space. It is used to generate the layout of the robot, assembly table, and parts boxes assuming that all components are rectangular. Two permutations Γ_+ and Γ_- are defined and a binary variable ϕ_i is introduced to represent the direction of component i. Given Γ_+, Γ_-, and ϕ_i, a unique layout can be generated. For a detailed explanation of the sequence-pair representation, please refer to Murata et al. (1996). In the following, we use an example to illustrate the sequence-pair representation for layout design.

4.1.1. Oblique grid

Consider a sequence pair (Γ_+, Γ_-) with $\Gamma_+ = (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11)$ and $\Gamma_- = (3, 7, 10, 1, 8, 5, 11, 2, 4, 0, 9, 6)$. The oblique grid for this sequence pair is a 12×12 grid. We label the vertical and horizontal lines of the grid with the component numbers in the sequence. Each point in this grid is the intersection of a vertical line and a horizontal line, which can be referred as (i, j). Then we rotate the grid by 45 degrees and put each component i at the point (i, i) in the rotated grid. See Fig. 2 for the resulted oblique grid.

4.1.2. Constraint graphs

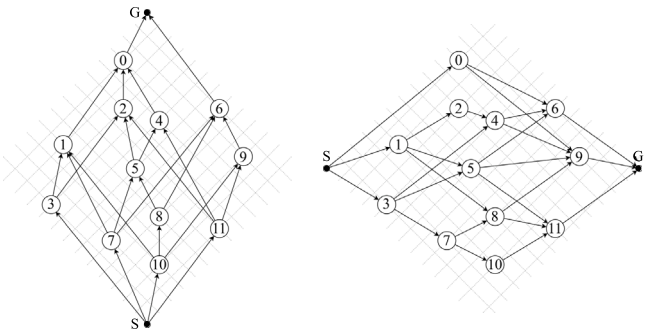
Given Γ_+ and Γ_-, the geometric relation of the components can be determined as follows:

- (Horizontal constraint): component i is to the left of component j if i is to the left of j in both Γ_+ and Γ_-.
- (Vertical constraint): component i is below component j if i appears after j in Γ_+ and before j in Γ_-.

Hence, when $\Gamma_+ = (\dots, i, \dots, j, \dots)$ and $\Gamma_- = (\dots, i, \dots, j, \dots)$, component i is to the left of component j. When $\Gamma_+ = (\dots, i, \dots, j, \dots)$ and $\Gamma_- = (\dots, j, \dots, i, \dots)$, component i is below component j. We can generate the horizontal vertical constraint graph based on the constraint relations between the components.

Let $G_h = (V_h, E_h)$ be the horizontal constraint graph and $G_v = (V_v, E_v)$ be the vertical constraint graph, where V_h and V_v are the set of vertices and E_h and E_v are the set of edges, respectively. The vertex set V_h includes the starting point s and the endpoint t, and all components. (s, i) and (i, t) are included in E_h, and $(i, j) \in E_h$ if i if and only if i is to the left of j according to the horizontal constraint. Similarly, the vertical constraint graph G_v can be generated. For our example, the horizontal and vertical constraint graphs are shown in Fig. 3.

The weight for each node is the width or length of the corresponding component in the corresponding direction. The starting point s and the endpoint t have a weight of 0.



(a) Vertical constraint graph. (b) Horizontal constraint graph.

Fig. 3. Given corresponding vertical constraint graph (a) and horizontal constraint graph (b). Directed edges between components represent constraints on the relative positions of those components.

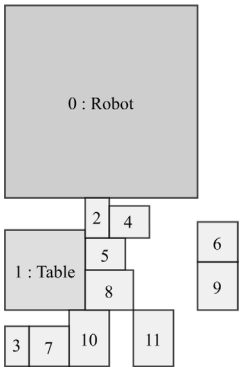


Fig. 4. Arrangement of the corresponding components.

4.1.3. Layout generation

Given the horizontal and vertical constraint graphs, the x-coordinate of a component i is obtained by solving the longest path problem from the source s to i in the horizontal constraint graph, and the y-coordinate of i can be computed by the vertical constraint graph. For the layout design problem with rotation, the parameter ϕ_i is used to determine whether the weight of the node is the width or length of the component. In our example, the layout of the components is shown in Fig. 4.

4.2. NSGA-II algorithm for optimal layout and motion planning

Recall that π represents the layout of a robotic cell, $f_{area}(\pi)$ is the area of the layout, and $f_{time}(\pi)$ is the total operation time of the robot arm. The following definitions are used to define the dominance of solutions.

Definition 1. A solution π_1 is said to dominate another solution π_2 if and only if the following conditions are satisfied:

- $f_{area}(\pi_1) \leq f_{area}(\pi_2)$ and $f_{time}(\pi_1) < f_{time}(\pi_2)$
- $f_{area}(\pi_1) < f_{area}(\pi_2)$ and $f_{time}(\pi_1) \leq f_{time}(\pi_2)$

Let P be a set of solutions, where $P = \{\pi_1, \pi_2, \ldots, \pi_n\}$. The non-dominated set P' is defined as follows:

Definition 2. Among a set of solutions P, the non-dominated set P' consists of those solutions that are not dominated by any other solutions in P.

When P include all the feasible solutions, the non-dominated set P' is called the Pareto-optimal set. The number of solutions of the layout design problem with n components is $(n!)^2$.

Theorem 1. *The number of solutions of the layout design problem is* $(n!)^2$.

The proof of Theorem 1 is omitted because it is straightforward. Since the search space is large, it is difficult to find the Pareto-optimal set by exhaustive search. Therefore, we use the NSGA-II (Non-dominated Sorting Genetic Algorithm II) algorithm to find a non-dominated set of solutions.

NSGA-II (Deb et al., 2002) is a multi-objective optimization algorithm that aims to find the Pareto front, which represents a set of non-dominated solutions in a multi-objective problem with multiple conflicting objectives.

To implement the NSGA-II algorithm for the integrated motion planning and layout design problem, an important assumption is that the operation time of the robot arm can be calculated by a function $f(z_{init}, z_{target}, C) = OT$ that returns the operation time OT for a robot arm to move from the initial posture z_{init} to the target posture z_{target} in the configuration space C. Furthermore, this study assumes that using the Rapidly exploring Random Trees Star (RRT*) algorithm can provide good approximations of the operation time for the robot arm given enough computation time.

The pseudo-code of NSGA-II is shown in Coello Coello et al. (2007). The algorithm first generates an initial population of candidate solutions. A candidate solution consists of the set of permutations of Γ_+, Γ_-, and binary variables ϕ to represent the layout in the sequence pair. After the generation of the layout candidate for each gene by sequence-pair, the motion planning algorithm is executed to obtain a collision-free path where the layout of the solution is fixed. We use a sampling-based motion planning algorithm RRT* (Karaman and Frazzoli, 2011) because the total operation time for a robot arm may be different according to the layout, initial and final postures, and environment. NSGA-II then applies genetic operators, such as crossover and mutation, to create offspring solutions from the parent solutions. Crossover combines the genetic information of two parent solutions to create two offspring solutions, while mutation introduces small changes in the genetic information of a solution. These genetic operators allow for the exploration of the solution space and the generation of diverse solutions.

NSGA-II uses a crowding distance measure to select solutions from the Pareto fronts for the next generation. The crowding distance reflects the density of solutions in the objective space, and it helps to maintain a diverse set of solutions by encouraging the selection of solutions that are farther apart from each other. In addition, NSGA-II carries over the best solution of the current generation to the next generation, so that the best solution is not lost in the process of evolution.

The order crossover is a genetic operator used in GA to generate offspring individuals from two-parent individuals. It selects a range of genes from the parent individuals and maintains their order while incorporating the remaining genes from the other parent to generate a new gene sequence. Specifically, two crossover points are randomly selected, and the gene sequence between these points is extracted and copied from parent 1 to the offspring. The remaining genes are then extracted from parent 2 in order, ignoring any genes that are already present in the offspring, and inserted into the empty spaces in sequence. The resulting offspring individual retains the characteristics of the parent individuals while having a new gene sequence. This crossover operator is an effective technique for expanding the search space in genetic algorithms. Mutation plays a key role in the exploration of the solution space by creating new solutions that are different from the parent solutions. For binary encoding, mutation flips the value of one or more bits in the binary string. It helps to introduce diversity in the population allowing the algorithm to escape local optima and search for new, potentially better solutions.

4.3. Motion planning algorithm

We use an RRT* algorithm (Karaman and Frazzoli, 2011), which is a commonly used motion planning method for robots, to obtain operation times. In RRT*, a group of neighboring nodes around the extended node is selected, and the node with the lowest cost is selected to become the parent node of the new node. First, the tree structure is built as in RRT*. Then, randomly sample points in the space and extend the tree in that direction from the nearest node. RRT* generates a set of neighboring nodes around the extended node and reconstructs the node with the lowest cost among them as the parent node. The radius for choosing neighbor nodes is determined by

$$r = \min\left(R\left(\frac{\log N}{N}\right)^{\frac{1}{d}}, \eta\right) \tag{20}$$

where N is the number of nodes, R is the weight, d is the number of the dimensions of the state space, and η is the upper bound of the radius. In this study, $R = 0.5$, $d = 7$ (6 axes + end-effector), and $\eta = 0.3$. The length of the edge to be extended at each step is 0.1[m], and the target posture z_{target} is selected as the sampling point with a probability of 20%. The joint angles at each time are obtained by RRT* with Open Motion Planning Library (OMPL). Then, the total operation time is computed from the time corresponding to the start and endpoints of the trajectory for each pick and place trajectory. Also, we include feasibility as a constraint condition since it is not feasible if the layout exceeds the maximum reach length of the robot. The RRT* algorithm is widely known and the time complexity of RRT* is generally considered to be $\mathcal{O}(N \log N)$ (Karaman and Frazzoli, 2011).

4.4. Surrogate-assisted NSGA-II

The NSGA-II with RRT* sometimes requires huge computing efforts because the RRT*-based motion planning is conducted for each individual to obtain the total operation time. To reduce the computational burden of the algorithm, a surrogate-assisted NSGA-II is proposed in this section.

In the proposed algorithm, the total operation time is estimated by a surrogate model to reduce the computation time of function evaluation in NSGA-II. To construct the surrogate model, we assume that the complex function $f(z_{init}, z_{target}, C) = OT$ can be approximated by a supervised learning model. In addition, since we use the RRT* algorithm to generate the training data, which is time-consuming, another assumption is that there is sufficient time to generate the training data. The surrogate model is trained by data generated from random sampling. Fig. 5 shows the framework and Algorithm 1 shows the pseudo-code of our proposed method.

In the first step, a surrogate model is created to estimate the robot's operation time. Using the RRT* algorithm, the exact robot operation times corresponding to the robot's initial and target postures are randomly sampled. Then, using sampled data as training data, a surrogate model is created by machine learning methods.

Then, the robot's operation time is estimated from the sequence pair from the given layout in the solution of NSGA-II. NSGA-II updates the generation using the area of the layout obtained and the total operation time of the robot. Steps 11–14 of Algorithm 2 are repeated pre-determined number of generation updates is reached. After that, the Pareto solution is obtained.

Because the Pareto solution is derived by using an estimated value from the surrogate model, the exact operation time and the robot's motion planning is obtained by the derived set of solutions using RRT*. The RRT* is executed multiple times for the derived solution. Eventually, the near-optimal layout design and the corresponding robot trajectory are obtained.

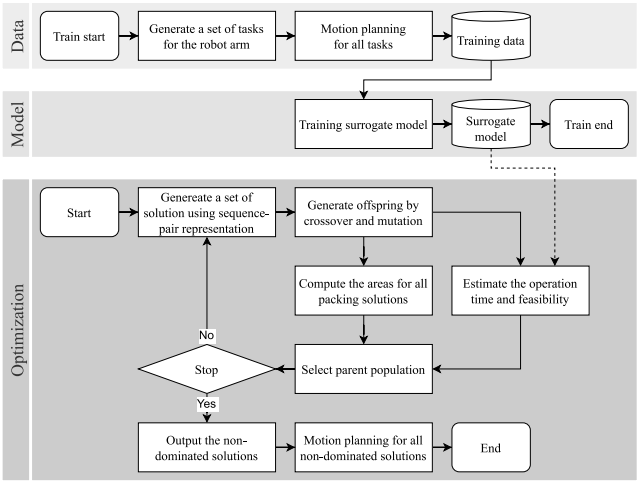


Fig. 5. Framework of the proposed algorithm.

Algorithm 1 Surrogate-assisted NSGA-II algorithm

Require: List of component sizes; Number of individuals $MaxPop$; Terminal generation number $MaxGen$; Configuration space C; Maximum computation time $MaxTime$; Number of samples $Samples$;
Ensure: Pareto-front[Floorplan π, Area f_{area}, Operation time f_{time}, Trajectory ζ]

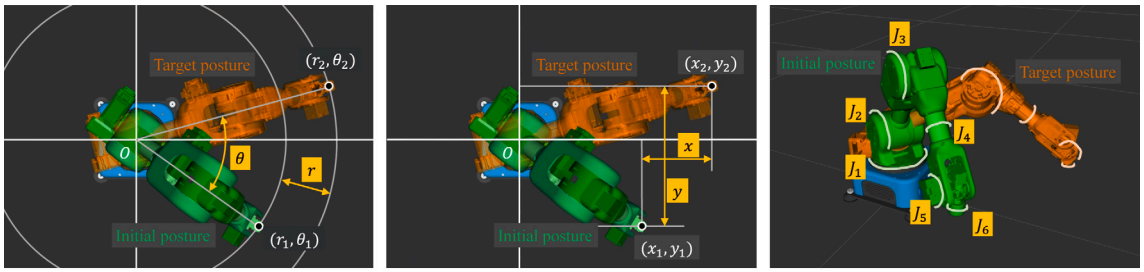
```
OperationTimeList ← Initialize();
for i = 1 : Samples do
    z_init, z_target ← RandomSampling(C);
    OT, F, ζ ← RRT*(C, z_init, z_target, MaxTime);
    OperationTimeList[i] ← OT;
end for
SurrogateModel ← GenModel(OperationTimeList);
ℙ ← InitializePopulation(MaxPop);
for i = 1 : MaxGen do
    f_1, f_2, c_1 ← ∅;
    for each p ∈ ℙ do
        π, f_area ← SequencePair(p);
        ℤ ← GetPickupAndPlacePosition(π);
        for each (z_init, z_target) ∈ ℤ do
            OT, F ← Predict(z_init, z_target, SurrogateModel);
            f_time ← f_time + OT;
            if F = False then
                break
            end if
        end for
        f_1 ← f_1 ∪ f_area; f_2 ← f_2 ∪ f_time; c_1 ← c_1 ∪ F;
    end for
    ℙ ← GenerationUpdate(ℙ, f_1, f_2, c_1);
    Pareto-front ← GetPareto(ℙ);
end for
Pareto-front, ζ ← Recalculate(Pareto-front);
return Pareto-front
```

4.5. Surrogate model

4.5.1. Selection of surrogate model

In building a surrogate model, training data is first created by random sampling. Here, the robot is given initial and target positions, RRT* is executed, and the robot's operation time is sampled. The following three types of initial and target positions given were used and compared (see Fig. 6).

- The difference between the $x - y$ coordinates of the robot's initial position and the target position (Fig. 6(a)).
- The difference between the $r - \theta$ coordinates of the robot's initial position and the target position (Fig. 6(b)).
- The difference of each joint angle $(J_1, J_2, J_3, J_4, J_5, J_6)$ in the initial and target posture of the robot (Fig. 6(c)).



(a) The polar coordinate system (r, θ) input to the surrogate model.

(b) The rectangular coordinate system (x, y) input to the surrogate model.

(c) The difference of the robot's joint angles $(J_1, J_2, J_3, J_4, J_5, J_6)$ input to the surrogate model.

Fig. 6. Three types of inputs when using the surrogate models. (a): the robot's initial and target postures are given, and the difference between r and θ, expressed in polar coordinates of the end-effector's position coordinates are used as input. (b): the difference between x and y, expressed in Cartesian coordinates for the position coordinates of the end-effector, is used as input. (c): the difference between the posture angles of each robot, $J_1, J_2, \ldots, J_6$ is used as input.

Table 2
Details of the robotic arm.

Item	Value
Manufacturer	Niryo
Model	Ned
Degrees of freedom	6
Reach	440 mm
Repeatability	±0.5 mm
Depth	200 mm
Width	200 mm

4.5.2. Surrogate models

We implemented and compared 7 surrogate methods: MLP (MultiLayer Perceptron), Ridge regression, Lasso regression, Elastic net, Random forest, GBDT (Gradient Boosting Decision Tree), SVR (Support Vector Regression) for the regression model. Details and parameters of each method are introduced in the following sections. Each method was implemented using Scikit-learn (Pedregosa et al., 2011).

5. Computational experiments

5.1. Experimental conditions

This section conducts computational experiments to evaluate the proposed method. To examine the robustness and adaptability of the proposed method, we conducted experiments under various conditions.

First, to build the surrogate model with high accuracy, we compared the accuracy of seven machine learning methods for the surrogate model, and we tested three types of feature sets for the surrogate model. Second, we compared the performance of our surrogate model with Izui's method (Izui et al., 2013), which has been intensively studied in the literature (Lim et al., 2017; Liu et al., 2024). Third, we compared the performance of the surrogate-assisted NSGA-II with the RRT*-based NSGA-II (Kawabe et al., 2022) and Izui's method in 10 different layout design problems.

We conducted our physical experiments using a 6-axis robotic arm, Niryo Ned. Fig. 7 shows the experimental system with a 6-axis robot arm. The details of this robot are shown in Table 2. Robot Operating System (ROS) Melodic with Ubuntu 18.04.6 LTS is used for the platform to Niryo Ned drivers: https://github.com/NiryoRobotics. The footprint of the Ned is 200 [mm] square, however, if a parts box is placed immediately next to the robot, the robot cannot reach the parts box, so a margin of 40 [mm] is provided.

Table 3 shows the heights and widths of the robots, tables, and parts boxes to be placed, and the order in which the parts boxes are placed.

Although Table 3 shows the sizes of 10 parts boxes, we varied the number of parts boxes in the experiments to examine the adaptability of the proposed method. Fig. 8 shows a robotic cell with one robot, one assembly table and five parts boxes.

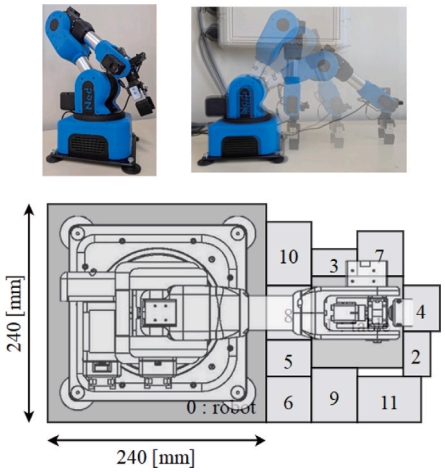


Fig. 7. Experimental system with Niryo Ned2 6-axis robotic arm.

Table 3
Size of components.

Component	i	Height [mm]	Width [mm]	Pick order
Robot	0	240	240	–
Assembly table	1	100	100	–
Parts box	2	30	50	1
	3	30	50	2
	4	40	50	3
	5	40	50	4
	6	50	50	5
	7	50	50	6
	8	60	50	7
	9	60	50	8
	10	70	50	9
	11	70	50	10

The number of individuals, the terminal generation number, the crossover rate and the mutation rate are set to 100, 500, 1.0, respectively, for the NSGA-II algorithm.

5.2. Comparison of surrogate models

We compared seven surrogate methods (MLP, Ridge regression, Lasso regression, Elastic net, Random forest, GBDT, SVR) for the regression model. Details and parameters of each method are introduced in Section 4.5.2. The surrogate model has three inputs: the difference between the initial robot hand position and the target position (polar and Cartesian coordinates), the difference between the initial robot posture and the target posture at each joint angle. The output of the surrogate model is the robot's operation time, which is the same for

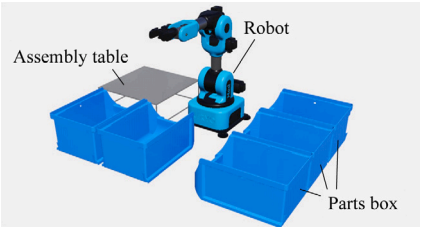


Fig. 8. A robotic cell.

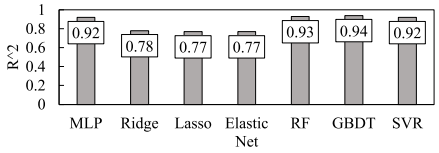


Fig. 9. Coefficient of determination R^2 for each method. The input to the regression model is the difference between the initial position of the robot hand (r_1, θ_1) and the target position (r_2, θ_2). The output is the robot's operation time.

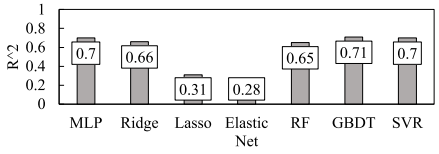


Fig. 10. Coefficient of determination R^2 for each method. The input to the regression model is the difference between the initial position of the robot hand (x_1, y_1) and the target position (x_2, y_2). The output is the robot's operation time.

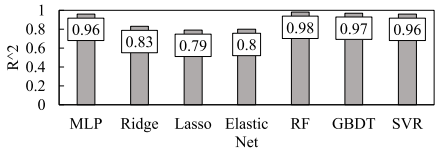


Fig. 11. Coefficient of determination R^2 for each method. The input of the regression model is the difference between the angles of each joint $(J_1, J_2, J_3, J_4, J_5, J_6)$ in the robot's initial posture and the angles $(J_1, J_2, J_3, J_4, J_5, J_6)$ in the target posture. The output is the robot's operation time.

all models. Figs. 9–11 show the coefficient of determination R^2 of the models created by each method. The number of samples was 10,000, of which 8000 were training data and 2000 were test data. It is clear that models such as MLP, Random Forest, and GBDT can construct predictive models with high accuracy from the results.

5.3. Performance of the proposed surrogate models

This subsection examines the accuracy of the surrogate model in estimating the robot's operation time. Methods such as mean squared error (MSE), mean absolute error (MAE), and coefficient of determination R^2 are commonly used to evaluate the performance of regression models. Since the interpretation of MSE and MAE depends on the scale of the data, the coefficient of determination R^2 is used to evaluate the performance of the surrogate model. The coefficient of determination of the surrogate model created was compared with that of conventional estimation models. The conventional estimation model in Eq. (1) developed by Izui et al. (2013) is used as the estimated operation time.

The proposed surrogate model and Eq. (1) were used to estimate the robot's operation time. Figs. 12–13 show the robot's operation time

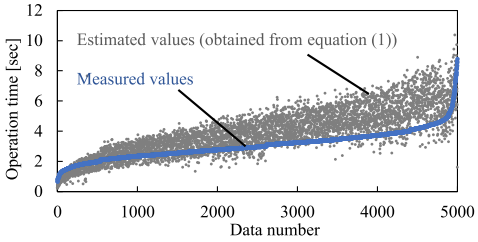


Fig. 12. Coefficient of determination R^2 of robot operation time using conventional estimation. Blue lines are the exact robot operation times, and gray dots are the operation times obtained using the estimation model. $R^2 = -0.80$.

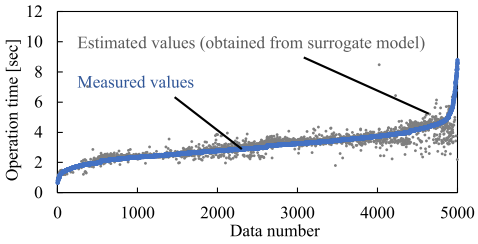


Fig. 13. Coefficient of determination R^2 of robot operation time derived by using surrogate models. The surrogate model with a random forest method. Blue lines are the exact robot operation times, and gray dots are the operation times obtained using the estimation model. $R^2 = 0.98$.

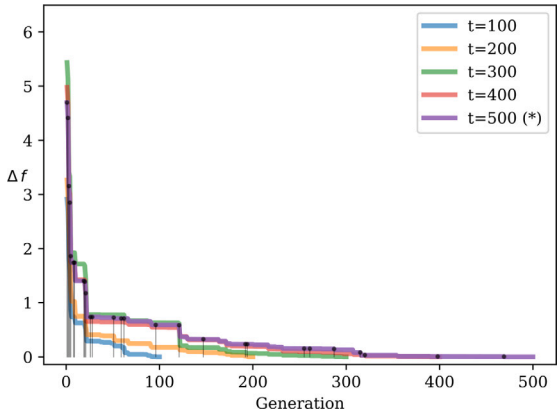


Fig. 14. Running Metrics for optimization calculations.

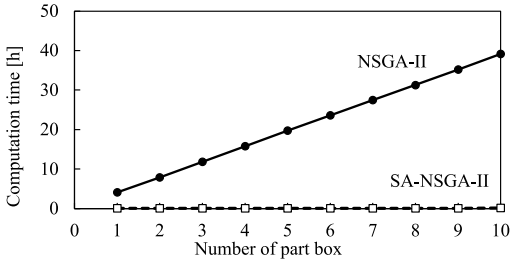


Fig. 15. Comparison of computation time for optimization calculations using RRT*-based NSGA-II and surrogate-assisted NSGA-II.

from 5000 random input values compared to the exact robot's operation times. The results show that the proposed surrogate model results in fewer measurement point errors than estimating the robot's operation time using Eq. (1). On the other hand, Eq. (1) cannot be accurately estimate the operating time.

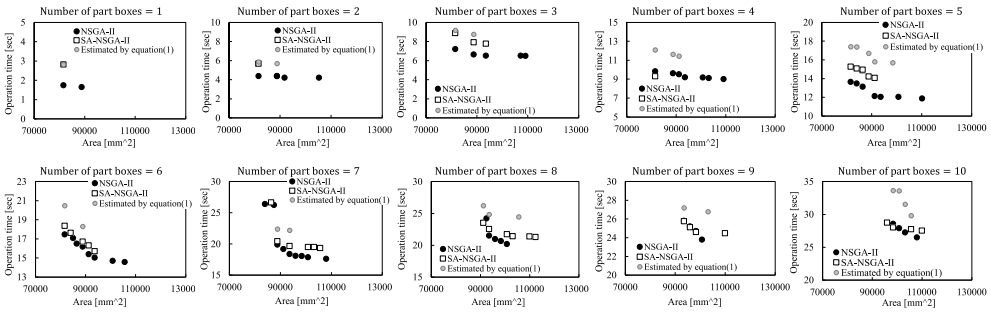


Fig. 16. Comparison of Pareto solution obtained using Eq. (1) for estimating the robot's operation time, RRT*-based NSGA-II considering the exact robot's operation time, and SA-NSGA-II using surrogate models under different numbers of parts boxes.

5.4. Performance of multi-objective optimization

5.4.1. Convergence analysis

The convergence of the Pareto front is evaluated by using Running Metric (Blank and Deb, 2020) for the analysis. Running Metric computes the Inverted Generational Distance (IGD) of each generation using a certain population of generations as a reference point set. We constructed a reference point set for every 100 generations. It then visualizes the degree of improvement using the survival rate of the algorithm.

IGD is a performance metric used in multi-objective optimization to evaluate the quality of a set of generated solutions compared to a known set of true Pareto-optimal solutions. A lower IGD value indicates better convergence and diversity of the generated solutions toward the true Pareto front, with values close to zero indicating higher quality solutions. Let $A = \{a_1, a_2, \ldots, a_{|A|}\}$ be the solution set obtained by the algorithm and $Z = \{z_1, z_2, \ldots, z_{|Z|}\}$ be the set of reference points, then IGD is calculated using the following equation.

$$IGD(A) = \frac{1}{|Z|}\left(\sum_{i=1}^{|Z|} \hat{d}_i^{\,p}\right)^{1/p} \tag{21}$$

where $\hat{d}_i$ is the Euclidean distance ($p = 2$) between z_i and the nearest reference point a_i from z_i.

Fig. 14 shows the Running Metric for the optimization calculations. The results show that the curves obtained when IGD is calculated for 400 generations are almost identical to those obtained when IGD is calculated for 500 generations.

5.4.2. Performance of surrogate-assisted NSGA-II

This section examines the performance of the proposed surrogate-assisted NSGA-II (SA-NSGA-II) in terms of computation time and the quality.

First, to evaluate the computation time, we compare the computation time of SA-NSGA-II and RRT*-based NSGA-II (NSGA-II) developed by Kawabe et al. (2022). Fig. 15 shows the changes in the respective computation times for optimization computations with SA-NSGA-II and RRT*-NSGA-II when the number of parts boxes is increased. The number of updated generations and the number of individuals per generation for NSGA-II are the same for both methods. The computation time for SA-NSGA-II includes the time required to recalculate the Pareto solution with RRT* after the optimization calculation using the surrogate model. The average of two experiments for each method and condition is obtained. The computational results show that the computation time for NSGA-II increases linearly with the number of parts boxes. The computation time for SA-NSGA-II is significantly shorter than that of the NSGA-II. This is due to the computational burden to calculate the motion planning of RRT* algorithm for all individuals to obtain an accurate operation time.

Second, we compare the quality of the Pareto solution obtained by SA-NSGA-II, NSGA-II, and Izui's approach (Izui et al., 2013). IGD represented by Eq. (21) is used to compare the Pareto solutions derived by SA-NSGA-II and NSGA-II. The solution set A in Eq. (21) is the set of Pareto solutions obtained by SA-NSGA-II, and the reference point set Z is the set of Pareto solutions obtained from NSGA-II. Table 4 shows the value of IGD for each number of parts boxes.

Table 4
IGD of the Pareto solution obtained by SA-NSGA-II and optimization by Eq. (1) as the reference point for the Pareto solution of NSGA-II.

Number of parts boxes	SA-NSGA-II	Estimated by Eq. (1)
1	0.6478	0.6485
2	0.3312	0.3255
3	0.1803	0.3106
4	0.1368	0.2290
5	0.1237	0.2386
6	0.0495	0.1185
7	0.0440	0.1530
8	0.0473	0.1360
9	0.0121	0.0867
10	0.0275	0.0923
Average	0.1600	0.2339

Fig. 16 shows all Pareto solutions obtained by each method for each case. From the results in Fig. 16, it can be seen that the Pareto solution is far from the optimal value when the operating time is estimated. On the other hand, Pareto solutions for SA-NSGA-II predicted by the surrogate model are close to Pareto solutions for NSGA-II, which uses accurate robot operation times. This is also confirmed by the smaller average value of IGD for SA-NSGA-II in Table 4. This is because the robot's operation time is precisely estimated using a highly accurate surrogate model, and the Pareto solution is recalculated by the motion planning after the NSGA-II computations.

This section compares the performance of the RRT*-based approach, surrogate-assisted approach, and the Izui's approach (Izui et al., 2013). The experimental results show that the Izui's approach has the worst performance in terms of accuracy and optimization results. The RRT*-based approach has the best performance in terms of accuracy and optimization results. However, it suffers from high computational cost. In addition, its computational cost increases rapidly as the number of parts boxes increases. This approach is suitable for small-scale problems and sufficient computational resources. On the other hand, the surrogate-assisted approach has a good balance between accuracy and computational cost. It can reduce the computational cost by 1/400 compared to the RRT*-based approach with higher accuracy. Furthermore, the experimental results show that the surrogate-assisted approach can find solutions that are comparable to the RRT*-based approach in a very short computation time. This approach is suitable for large-scale problems and limited computational resources.

6. Conclusion

A surrogate-assisted-NSGA-II (SA-NSGA-II) has been proposed for simultaneous layout design and motion planning for RCMSs. Computational experiments show that the surrogate model has sufficient

accuracy with 98% of the robot's operation time. The SA-NSGA-II can reduce the 40 h of computation time into seconds while the IGD between the exact Pareto solution is 0.1715. The proposed SA-NSGA-II can obtain the Pareto solutions within 1/400 of computation time, which is significantly less computation time than the conventional RRT*-based NSGA-II with higher accuracy. The findings of this study have a number of important implications for future practice. Several limitations and future research directions of the present study should be noted. First, this study focuses on the single robot system for the pick and place operation in the RCMSs. By applying the proposed method to RCMSs with multiple robots, the efficiency and feasibility of the manufacturing systems can be improved. It is interesting to explore how to implement multi-robot systems while avoiding collisions and improving the efficiency of the manufacturing system. One of the key issues is that it is necessary to develop an efficient motion and path planning algorithm to solve a conflict-free motion planning problem within a limited computation time. Second, this study assumes the manufacturing system is deterministic. However, the manufacturing system is often stochastic due to uncertainty of the order, the processing time, and the machine failure. In the future, discussing the dynamic cellular manufacturing system (DCMS) with the proposed method is important.

CRediT authorship contribution statement

Tomoya Kawabe: Writing – original draft, Visualization, Validation, Software, Formal analysis, Data curation. **Tatsushi Nishi:** Writing – review & editing, Supervision, Resources, Project administration, Investigation, Funding acquisition, Conceptualization. **Ziang Liu:** Writing – review & editing, Writing – original draft, Methodology, Data curation, Conceptualization. **Tomofumi Fujiwara:** Writing – original draft, Software, Methodology.

Declaration of competing interest

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.

Acknowledgments

This research is subsidized by New Energy and Industrial Technology Development Organization (NEDO) under a project JPNP20016. This paper is one of the achievements of joint research with ROBOT Industrial Basic Technology Collaborative Innovation Partnership (ROBOCIP) and ROBOCIP has a joint copyright of this paper. The authors would like to thank the funding provided by the project.

Data availability

Data will be made available on request.

References

Akturk, M.S., Turkcan, A., 2000. Cellular manufacturing system design using a holonistic approach. Int. J. Prod. Res. 38 (10), 2327–2347.
Bartz-Beielstein, T., Zaefferer, M., 2017. Model-based methods for continuous and discrete global optimization. Appl. Softw. Comput. 55, 154–167.
Bayram, H., Şahin, R., 2016. A comprehensive mathematical model for dynamic cellular manufacturing system design and linear programming embedded hybrid solution techniques. Comput. Ind. Eng. 91, 10–29.
Blank, J., Deb, K., 2020. A running performance metric and termination criterion for evaluating evolutionary multi- and many-objective optimization algorithms. In: 2020 IEEE Congress on Evolutionary Computation. CEC, pp. 1–8.
Calisto, M.B., Lai-Yuen, S.K., 2021. EMONAS-Net: Efficient multiobjective neural architecture search using surrogate-assisted evolutionary algorithm for 3D medical image segmentation. Artif. Intell. Med. 119, 102154.
Chang, Y.C., Chang, Y.W., Wu, G.M., Wu, S.W., 2000. B*-trees: A new representation for non-slicing floorplans. In: Proceedings of the 37th Design Automation Conference. pp. 458–463.
Chugh, T., Chakraborti, N., Sindhya, K., Jin, Y., 2017. A data-driven surrogate-assisted evolutionary algorithm applied to a many-objective blast furnace optimization problem. Mater. Manuf. Process. 32 (10), 1172–1178.
Coello Coello, C.A., Lamont, G.B., Van Veldhuizen, D.A., 2007. Evolutionary Algorithms for Solving Multi-Objective Problems. Springer US.
Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. 6 (2), 182–197.
Feng, H., Xi, L., Xia, T., Pan, E., 2018. Concurrent cell formation and layout design based on hybrid approaches. Appl. Soft Comput. 66, 346–359.
Forghani, K., Fatemi Ghomi, S.M.T., 2020. Joint cell formation, cell scheduling, and group layout problem in virtual and classical cellular manufacturing systems. Appl. Soft Comput. 97, 106719.
Forghani, K., Fatemi Ghomi, S.M.T., Kia, R., 2020. Solving an integrated cell formation and group layout problem using a simulated annealing enhanced by linear programming. Soft Comput. 24 (15), 11621–11639.
Helms, M.M., 2021. Encyclopedia of management.
Izui, K., Murakumo, Y., Suemitsu, I., Nishiwaki, S., Noda, A., Nagatani, T., 2013. Multiobjective layout optimization of robotic cellular manufacturing systems. Comput. Ind. Eng. 64 (2), 537–544.
Jin, Y., 2011. Surrogate-assisted evolutionary computation: recent advances and future challenges. Swarm and Evolutionary Computation 1 (2), 61–70.
Karaman, S., Frazzoli, E., 2011. Sampling-based algorithms for optimal motion planning. Int. J. Robot. Res. 30 (7), 846–894.
Karimi-Mamaghan, M., Mohammadi, M., Meyer, P., Karimi-Mamaghan, A.M., Talbi, E.-G., 2022. Machine learning at the service of meta-heuristics for solving combinatorial optimization problems: A state-of-the-art. European J. Oper. Res. 296 (2), 393–422.
Kawabe, T., Liu, Z., Nishi, T., Alam, M.M., Fujiwara, T., 2022. Optimal motion planning and layout design in robotic cellular manufacturing systems. In: 2022 IEEE International Conference on Industrial Engineering and Engineering Management. pp. 1541–1545.
Kawabe, T., Nishi, T., Liu, Z., Fujiwara, T., 2024. Task planning for robot manipulator using natural language task input with large language models. In: 2024 IEEE 20th International Conference on Automation Science and Engineering. pp. 3484–3489.
Kia, R., Baboli, A., Javadian, N., Tavakkoli-Moghaddam, R., Kazemi, M., Khorrami, J., 2012. Solving a group layout design model of a dynamic cellular manufacturing system with alternative process routings, lot splitting and flexible reconfiguration by simulated annealing. Comput. Oper. Res. 39 (11), 2642–2658.
Kozik, A., 2017. Handling precedence constraints in scheduling problems by the sequence pair representation. J. Comb. Optim. 33 (2), 445–472.
Leiber, D., Elckholt, D., Vuong, A.T., Reinhart, G., 2022. Simulation-based layout optimization for multi-station assembly lines. J. Intell. Manuf. 33, 537–554.
Li, X., Liu, M., Li, S., 2022. A surrogate-assisted evolutionary algorithm with an adaptive sample selection strategy for wind farm layout optimization. Int. J. Green Energy 1–12.
Lim, Z.Y., Ponnambalam, S.G., Izui, K., 2016. Nature inspired algorithms to optimize robot workcell layouts. Soft Comput. 49, 570–589.
Lim, Z.Y., Ponnambalam, S.G., Izui, K., 2017. Multi-objective hybrid algorithms for layout optimization in multi-robot cellular manufacturing systems. Knowl.-Based Syst. 120, 87–98.
Liu, Z., Kawabe, T., Nishi, T., Ito, S., Fujiwara, T., 2024. Surrogate-assisted multi-objective optimization for simultaneous three-dimensional packing and motion planning problems using the sequence-triple representation. Appl. Artif. Intell. 38 (1).
Liu, Q., Meller, R.D., 2007. A sequence-pair representation and MIP-model-based heuristic for the facility layout problem with rectangular departments. IIE Trans. 39 (4), 377–394.
Liu, Z., Nishi, T., 2024. Surrogate-assisted evolutionary optimization for perishable inventory management in multi-echelon distribution systems. Expert Syst. Appl. 238, 122179.
Long, H., Li, W., 2020. A data-driven evolutionary algorithm for wind farm layout optimization. Energy 208, 118310.
Meller, R.D., Chen, W., Sherali, H.D., 2007. Applying the sequence-pair representation to optimal facility layout designs. Oper. Res. Lett. 35 (5), 651–659.
Mikyu, N., Nishi, T., Liu, Z., Fujiwara, T., 2024. Three-dimensional bin packing problems with the operating time of a robot manipulator. IFIP Adv. Inf. Commun. Technol. 44–60.
Mohammadi, M., Forghani, K., 2019. Solving a stochastic berth allocation problem using a hybrid sequence pair-based simulated annealing algorithm. Eng. Optim. 51 (10), 1810–1828.
Murata, H., Fujiyoshi, K., Nakatake, S., Kajitani, Y., 1996. VLSI module placement based on rectangle-packing by the sequence-pair. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. 15 (12), 1518–1524.
Nagele, L., Hoffmann, A., Schierl, A., Reif, W., 2020. LegoBot: Automated planning for coordinated multi-robot assembly of LEGO structures. In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 9088–9095.

Nonoyama, K., Liu, Z., Fujiwara, T., Alam, M.M., Nishi, T., 2022. Energy-efficient robot configuration and motion planning using genetic algorithm and particle swarm optimization. Energies 15 (6), 2074.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in python. J. Mach. Learn. Res. 12, 2825–2830.

Salimpour, S., Pourvaziri, H., Azab, A., 2021. Semi-robust layout design for cellular manufacturing in a dynamic environment. Comput. Oper. Res. 133, 105367.

Shiller, Z., 1989. Interactive time optimal robot motion planning and work-cell layout design. In: Proc. 1989 International Conference on Robotics and Automation. pp. 964–969.

Suemitsu, I., Izui, K., Yamada, T., Nishiwaki, S., Noda, A., Nagatani, T., 2016. Simultaneous optimization of layout and task schedule for robotic cellular manufacturing systems. Comput. Ind. Eng. 102, 396–407.

Togo, H., Asanuma, K., Nishi, T., Liu, Z., 2022. Machine learning and inverse optimization for estimation of weighting factors in multi-objective production scheduling problems. Appl. Sci. 12, 9472.

Tong, H., Huang, C., Minku, L.L., Yao, X., 2021. Surrogate models in evolutionary single-objective optimization: A new taxonomy and experimental study. Inf. Sci. 562, 414–437.

Wang, H., Jin, Y., 2021. A random forest-assisted evolutionary algorithm for data-driven constrained multiobjective combinatorial optimization of trauma systems. IEEE Trans. Cybern. 50 (2), 536–549.

Zhang, J., Fang, X., 2013. Response surface method based robotic cells layout optimization in small part assembly. In: Proc. IEEE ISR 2013. pp. 1–6.

Zhang, J., Li, X., Tian, J., Jiang, Y., Luo, H., Yin, S., 2023a. A variational local weighted deep sub-domain adaptation network for remaining useful life prediction facing cross-domain condition. Reliab. Eng. Syst. Saf. 231, 108986.

Zhang, J., Tian, J., Alcaide, A.M., Leon, J.I., Vazquez, S., Franquelo, L.G., Luo, H., Yin, S., 2023b. Lifetime extension approach based on the levenberg–marquardt neural network and power routing of DC-DC converters. IEEE Trans. Power Electron. 38 (8), 10280–10291.

Zhang, M., Yan, J., 2021. A data-driven method for optimizing the energy consumption of industrial robots. J. Clean. Prod. 285, 124862.