# SmartFixture: Physics-guided reinforcement learning for automatic fixture layout design in manufacturing systems

Yinan Wang, Tim Lutz, Xiaowei Yue & Juan Du

**To cite this article:** Yinan Wang, Tim Lutz, Xiaowei Yue & Juan Du (11 Oct 2024): SmartFixture: Physics-guided reinforcement learning for automatic fixture layout design in manufacturing systems, IISE Transactions, DOI: 10.1080/24725854.2024.2401041

**To link to this article:** https://doi.org/10.1080/24725854.2024.2401041

IISE TRANSACTIONS
https://doi.org/10.1080/24725854.2024.2401041

# SmartFixture: Physics-guided reinforcement learning for automatic fixture layout design in manufacturing systems

Yinan Wang[a], Tim Lutz[b], Xiaowei Yue[c], and Juan Du[d]

[a]Rensselaer Polytechnic Institute, Troy, NY, USA; [b]The University of South Carolina, Columbia, SC, USA; [c]Tsinghua University, Beijing, China; [d]The Hong Kong University of Science and Technology, Hong Kong, China

**ABSTRACT**
Fixture layout design critically impacts the shape deformation of large-scale sheet parts and the quality of the final product in the assembly process. The existing works focus on developing Mathematical-Optimization (MO)-based methods to generate the optimal fixture layout via interacting with Finite Element Analysis (FEA)-based simulations or its surrogate models. Their limitations can be summarized as memorylessness and lack of scalability. Memorylessness indicates that the experience in designing the fixture layout for one part is usually not transferable to others. Scalability becomes an issue for MO-based methods when the design space of fixtures is large. Furthermore, the surrogate models might have limited representation capacity when modeling high-fidelity simulations. To address these limitations, we propose a learning-based framework, SmartFixture, to design the fixture layout by training a Reinforcement learning agent through direct interaction with the FEA-based simulations. The advantages of the proposed framework include: (i) it is generalizable to design fixture layouts for unseen scenarios after offline training; (ii) it is capable of finding the optimal fixture layout over a massive search space. Experiments demonstrate that the proposed framework consistently generates the best fixture layouts that receive the smallest shape deformations on the sheet parts with different initial shape variations.

**ARTICLE HISTORY**
Received 21 June 2023
Accepted 11 August 2024

**KEYWORDS**
Fixture layout design; reinforcement learning; quality assurance

## 1. Introduction

With the emerging need for highly customized products, the flexible design of the manufacturing process has become a critical research problem for product quality assurance. For example, fixture layout design in the manufacturing process directly determines the shape deformation of the final products, especially for those products assembled by large-scale sheet metal or composite parts. The large-scale sheet parts usually feature a small ratio between their thickness and length or width and have been widely used in manufacturing ship hulls (Du *et al.*, 2021), marine structures (Parks, 2020), automobiles (Pavlović *et al.*, 2020), aircraft (Shroff *et al.*, 2017), etc. However, they are prone to shape deformations caused by external loading or even gravity. Fixtures are generally used in the manufacturing process to locate and support different parts. Thus, appropriately adjusting the fixture layout for different parts and products can effectively reduce the shape deformation of sheet metal or composite parts and further improve the quality of final products.

In general, fixture layout designs for deformable sheet parts follow the "$N-2-1$" principle ($N \geq 3$) (Cai *et al.*, 1996). The $N$ fixtures are usually placed under the sheet part to support it and have a significant influence on its shape deformation. The challenges of fixture layout design mainly lie in three aspects:

1. Deploying tens of fixtures over thousands of candidate locations creates a massive design space, which significantly raises the computational cost.
2. It can be difficult to formulate the shape deformation or other quality metrics accurately and adaptively in the manufacturing process by an explicit expression (e.g., physical models).
3. Fixture deployment is repeated from scratch for heterogeneous products, which causes duplicated computational costs.

The problem of fixture layout design is usually formulated as finding the best locations of these $N$ fixtures via the interaction between a "solver" and an "environment". Intuitively speaking, the role of the "solver" is to strategically generate different layouts of fixtures, and the role of the "environment" is to take a specific layout to obtain the corresponding shape deformation. The development of novel "solver" and "environment" in existing literature aims to mitigate or resolve the aforementioned challenges. Various types of optimization-based methods, such as mathematical programming, heuristic optimization, etc., have been

CONTACT Xiaowei Yue yuex@tsinghua.edu.cn
Supplemental data for this article is available online at10.1080/24725854.2024.2401041.

proposed as the "solver" (Huang *et al.*, 2010; Zhong *et al.*, 2023). Although these methods have been specifically tailored to improve computational efficiency, their computational costs still grow exponentially with the increase of design space. Furthermore, optimization-based methods are usually memoryless and lack scalability, which requires them to repeat the optimization from scratch for each heterogeneous product, and thus unable to transfer knowledge into unseen scenarios.

Learning-based methods have been recently developed to solve the optimization problem with better generalizability and the ability to transfer knowledge. It also demonstrated outstanding performance in dealing with massive search space. For example, Deep Reinforcement Learning (DRL) has been developed to master the game of Go (AlphaGo in 2016 and AlphaZero in 2018) and beat the human experts (Silver *et al.*, 2016; Silver *et al.*, 2018), in which the DRL learns the principle of the game of Go by interacting with the "environment", and can further generalize the knowledge into practice. Compared with basic model-based reinforcement learning, DRL enables high dimensional state and action spaces, which prepare it for a complex and massive search space. However, there is limited research work developing DRL-based methods to optimize the design of the manufacturing process (i.e., fixture layout for sheet metal or composite parts). One trial is to apply DRL to design fixtures for rigid-body products (Low *et al.*, 2020). However, this work is formulated to place the fixtures over all the pre-selected candidate locations and then iteratively eliminate some of them. Furthermore, it assumes the product has an ideal shape, and the objective is simply to fix the product, which may not work well in practice.

Although the manufacturing process shares similar characteristics to the Go game (i.e., massive search space), it is uniquely governed by physical principles. There is an urgent need to shed light on how to borrow the idea of AlphaGo for the design of the manufacturing process. To fill in this research gap, we propose a physics-guided DRL framework, SmartFixture, for the automatic fixture layout design in manufacturing systems. The proposed framework formulates the fixture layout design as a Markov decision process and then solves it using the policy improvement algorithm. The advantages of the proposed SmartFixture can be summarized into three aspects:

1. SmartFixture provides the first learning-based framework for optimizing fixture layout in manufacturing systems by directly interacting with finite element analysis-based simulations.
2. It is highly scalable to process high-dimensional input data and optimize over a massive search space.
3. It has an outstanding generalization ability to transfer knowledge into unseen scenarios without further training.

The remainder of this article is organized as follows. Section 2 reviews the existing research works on fixture layout design. Section 3 introduces each building block in the proposed SmartFixture in detail and summarizes the training algorithm. Section 4 investigates the performance of the proposed framework on reducing the shape deformation of large-scale sheet metal or composite parts with or without initial shape deformations. Finally, Section 5 concludes this article. Although in the case study, we mainly focus on sheet metal or composite parts in a rectangular shape, the proposed approach can also be applicable to other types of parts.

## 2. Literature review

The existing literature on fixture layout design can be categorized according to different methods developed as the "solver" and "environment", which are summarized in Table 1. It also clearly distinguishes our proposed method from existing ones, as it is the first trial to exploit learning-based optimization on fixture layout design. A detailed literature review is introduced as follows.

Mathematical Optimization (MO)-based have been one type of widely developed "solvers" for fixture layout design, and Finite Element Analysis (FEA) is most often used to build the "environment" to simulate the manufacturing process. In this setting, heuristic optimization methods have been developed because of their strength in solving highly nonlinear optimization problems. Menassa and DeVries (1991) proposed the Broyden–Fletcher–Goldfarb–Shanno optimization algorithm to determine the fixture support locations and reduce the workpiece deflection caused by assembly or machining loads. Kulankara *et al.* (2001) further proposed to apply a genetic algorithm to simultaneously optimize fixture layout and clamping force for a compliant workpiece. However, the regularly used heuristic algorithms are easily trapped in local optima. To tackle this problem, Xing (2017) proposed to apply global optimization algorithms to generate the fixture layout for sheet metal assemblies. In addition to the heuristic optimization algorithms, nonlinear programming methods have also been developed as "solvers". Cai *et al.* (1996) proposed a nonlinear programming method to

**Table 1.** Literature review on fixture layout design.

| *Solver\Env.* | *Surrogate Model* | *System Equation* | *FEA-based Simulation* |
|---|---|---|---|
| Heuristic Optimization | Kim and Ding (2004)<br>Yang *et al.* (2017) | Du *et al.* (2021) | Menassa and DeVries (1991)<br>Kulankara *et al.* (2001)<br>Xing (2017) |
| Nonlinear Optimization | Yue and Shi (2018)<br>Yue *et al.* (2018)<br>Du *et al.* (2019) | Mou *et al.* (2023)<br>Zhong *et al.* (2023) | Cai *et al.* (1996)<br>Camelio *et al.* (2004)<br>Huang *et al.* (2010) |
| Learning-based Optimization | | | **SmartFixture** (ours) |

design the fixture layout through the interaction with FEA simulation to minimize the part deformation under a given force. Camelio *et al.* (2004) further extended the optimization for a single part into reducing the shape variations for the final assembled product using nonlinear programming. The sequential space-filling method is also adapted to generate the fixture layout over the two-dimensional design space (Huang *et al.*, 2010). However, this line of research works has a common issue of high computational cost. The reasons can be summarized as (i) the computational complexity in optimizing the fixture layout grows exponentially with the increase of search space; (ii) these methods do not have the mechanisms to generalize the knowledge into unseen scenarios. Thus, whenever a new scenario appears, it usually repeats from scratch and repetitively requires a massive amount of simulation from the FEA-based "environment", which causes duplicated computational costs.

To tackle this problem, one idea is to improve the computational efficiency in generating simulations from the FEA-based "environment". Surrogate models are proposed to approximate and replace the FEA methods in serving as the "environment". The general idea is to build a high-accuracy surrogate model to approximate the output from the manufacturing process and then use it to replace the FEA method. For example, a linear state-space variation model was built to link the fixture layout to the product variation, and an improved basic exchange algorithm was developed to generate the fixture layout (Kim and Ding, 2004). Similarly, a Kriging model was built as a surrogate by training on the simulations collected from the FEA method, and the cuckoo search algorithm (a meta-heuristic algorithm) was then applied to generate the optimal fixture layout by Yang *et al.* (2017). Yue and Shi (2018) proposed a grouped Latin Hypercube Sampling approach to guide the data collection from the FEA method for training a universal Kriging model and then developed an optimal feed-forward control algorithm to optimize the control actions to reduce the shape deviations. Yue *et al.* (2018) further proposed an Automatic Optimal Shape Control (AOSC) system for large-scale composite parts by considering different sources of uncertainties in the assembly process and jointly exploiting FEA simulation, design of experiment (DOE), surrogate modeling, and multivariable optimization. To improve the optimization method, Du *et al.* (2019) extended the framework by incorporating the idea of sparse learning into the optimization process. To improve the performance of the surrogate model, Yue *et al.* (2021) first exploited active learning in building the Gaussian process considering different sources of uncertainties in the assembly process, and Lee *et al.* (2022) further extended the Gaussian process by proposing a neural network Gaussian process considering input uncertainty. Bayesian optimization is further developed to exploit the established surrogate model for the shape control of a fuselage (AlBahar *et al.*, 2024; Wang and Yue, 2024). However, the "environment" built upon surrogate models still raises concerns about its accuracy, especially for those scenarios that are not well represented in the training data.

In addition to the surrogate model, the linearized FEA system equation is another line of research work that both improves the computational efficiency and preserves the accuracy of the FEA-based "environment". The basic idea is to directly use the governing system equation to replace the entire FEA model to serve as the "environment". For example, the global stiffness matrix and global load vector are exported from the FEA model and fed into the direct stiffness method to efficiently calculate the shape deformation, and a binary integer programming approach is formulated to optimize the fixture layout (Du *et al.*, 2021). Mou *et al.* (2023) further utilized the feedback information to update the shape control model in an online manner to resolve the mismatch between the digital model and the physical model. Zhong *et al.* (2023) integrated the system equation with the convex relaxation method. Although utilizing the system equation to replace the FEA method achieves a good trade-off between computational efficiency and accuracy, its scalability is limited, as it is not intuitive to export the system equations of the FEA when a more complex simulation is required to better represent the physical world.

With the development of high-fidelity simulation, the digital twin concept attracted increasing attention from researchers, due to its ability to simulate the real manufacturing process or even the entire manufacturing system (Wen *et al.*, 2018; Lutz *et al.*, 2022; Zhong *et al.*, 2022). Digital twins can serve as the close-to-reality "environment" to benefit the design and optimization of the manufacturing process. However, the complexity of a digital twin can make it challenging to approximate it by surrogate models or system equations. Furthermore, the development of the digital twin also brings a massive search space, which is challenging for the existing optimization-based methods. Thus, the digital twin crates interest in improving the "solver" to make it directly interact with the digital twin and create scalability and generalizability.

Compared with optimization-based methods, learning-based methods have good scalability and generalizability to serve as the "solver", but are yet to be well investigated in solving the fixture layout design problem. The major difference between learning-based methods and optimization-based methods is that the learning-based methods are designed to gain experience on existing scenarios and can generalize the knowledge into unseen scenarios. This property naturally enables the learning-based methods to be applied in an online manner after sufficient offline training, which eliminates the duplicated computational costs on heterogeneous products. Furthermore, learning-based methods have the ability to efficiently deal with a high-dimensional search space, especially with the development of deep learning methods (He *et al.*, 2016). Amongst the learning-based methods developed for optimization problems, Reinforcement Learning (RL) has demonstrated an outstanding performance and capability to interact with complex "environments" to optimize the objective function in a large-scale search space. RL-based methods have been widely developed to master a large variety of tasks, including playing games (Silver *et al.*, 2016; Silver *et al.*, 2018), chip design (Mirhoseini *et al.*, 2021), autonomous driving (Kiran *et al.*, 2021), and optimizing oil and gas field development plans (Nasir *et al.*, 2021; He *et al.*, 2022), among others. Its application in manufacturing systems has started to attract the interest of researchers, such as the RL for process control and defect mitigation (Chung *et al.*, 2022) and multi-robot fixture planning (Canzini *et al.*, 2024). Both approaches achieved very promising performance in manufacturing systems. However, research gaps still exist in developing RL-based methods for the design and optimization of manufacturing processes, due to the complex system characteristics.

## 3. SmartFixture: A physics-guided RL methodology

In this section, a physics-guided RL framework, SmartFixture, is proposed for automatic fixture layout design in a manufacturing system. We first give an overview of the proposed framework wherein we formulate the automatic fixture layout design as a Markov Decision Process (MDP) that can then be solved with RL. Then, each module in the proposed framework is introduced in detail.

### *3.1. SmartFixture framework overview*

The objective of the proposed SmartFixture is to provide a general framework to automatically design the layout of fixtures to reduce shape deviation (e.g., the deformation caused by gravity, the gap between adjacent parts, etc.) in manufacturing process. Such a learning-based framework could (i) transfer knowledge both from the digital twin to the real manufacturing process and from the manufacturing process of one product to its similar products and (ii) solve the large-scale design problem for manufacturing systems. Thus, the development cycle of a new product can be significantly reduced.

The process of fixture layout design can be formulated as an MDP and solved by RL in a setup as visualized in Figure 1. An FEA simulation inside the ANSYS simulation software (Madenci and Guven, 2015) is directly used as the environment to model the physical process accurately. It is connected to a Deep Neural Network (DNN) that acts as an RL agent, which learns how to design the fixture layout by interacting with, and learning from, the environment. In general, an MDP is a discrete process, and at each time step $t$, the environment will generate the observation of current state $s_t$, and the agent selects the action $a_t$ according to the current policy $\pi_\theta(a_t|s_t)$, which is a probability distribution over the action space $\mathcal{A}$ conditioned on the observed state $s_t$. The environment evolves to the next state $s_{t+1}$ by taking the selected action $a_t$, and the reward $r_t$ returned by this action is generated accordingly. Thus, the MDP can be denoted as a sequence of tuples $(s_t, a_t, r_t)$. An important assumption in MDP is the Markov property (Howard, 1960), which means the transition to the new state $s_{t+1}$ only depends on its previous state $s_t$.

To formulate the fixture layout design as an MDP, we initialize the environment with no fixture and add one new fixture at each time step until the number of fixtures reaches the control limit. The objective is to automatically design the fixture layout to minimize the deformation of the large-scale sheet part. The state is denoted as $s_t$ (Section 3.4), which stores the output from the "environment" (Section 3.2), including the layout of existing fixtures, the current shape deformation, and the internal stress given the existing fixtures. The reward is designed to reflect the effect of the entire set of fixtures (Section 3.3). A larger reward indicates a smaller final deformation. The action $a_t$ denotes the position of the newly added fixture (Section 3.5).

### *3.2. Environment*

We employ the Finite Element Method (FEM) (Szabó and Babuška, 2021) to build an environment that simulates the deformation of a large-scale sheet part under different fixture layouts. ANSYS (Madenci and Guven, 2015) is our chosen simulation platform, as it provides an interface through which the RL agent can communicate with the simulation. This is accomplished by utilizing the Mechanical

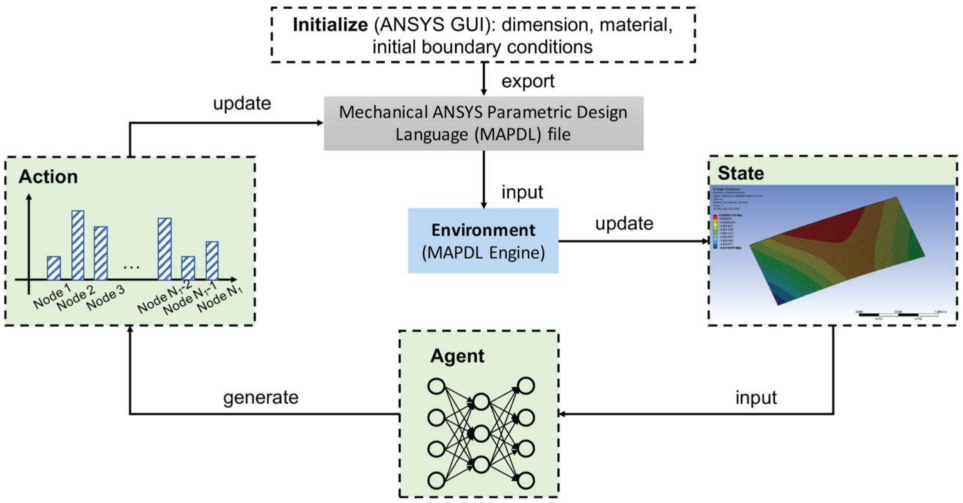
**Initialize** (ANSYS GUI): dimension, material, initial boundary conditions
export
update
Mechanical ANSYS Parametric Design Language (MAPDL) file
input
Action
Node 1
Node 2
Node 3
…
Node N₁-2
Node N₁-1
Node N₁
Environment (MAPDL Engine)
update
State
Agent
generate
input

Figure 1. Overview of the entire framework.

ANSYS Parametric Design Language (MAPDL) (Kaszynski, 2020). Thus, the RL agent can directly receive the output from the simulation as the current state and generate the corresponding action to, in turn, update the simulation for the next time step. We consider two types of sheet parts in the simulation using different materials. One is a flat sheet part with the dimension of $2\ \text{m} \times 1\ \text{m}\ \times 0.002\ \text{m}$ (length, width, thickness) using the structural steel that is commonly used in ship panel assembly (Du *et al.*, 2021). The other is a fleet sheet part with the dimension of $2\ \text{m}\ \times 1\ \text{m} \times 0.004\ \text{m}$ (length, width, thickness) using a composite material, which is made of four plies, and the orientations of the four plies are 90 deg/0 deg/90 deg/0 deg. This specification follows the setup in Zhang and Shi (2016) and Yue and Shi (2018). After generating a mesh over the target object, all finite element nodes on the bottom surface are candidate locations for the fixtures. Initial deformations inevitably exist in real-life sheet metal or composite parts, so our simulation is designed to generate various initial deformations to mimic this situation. To do so, our sheet metal or composite parts are fixed at their center node, and vertical displacements are applied at the four corners to generate random deformations, as demonstrated in Figure 2. The displacements at each corner are sampled from $[-0.002\ \text{m}, 0.002\ \text{m}]$ according to the Latin Hypercube Design (LHD) (Tang, 1993), thus introducing warpage to the part. The resulting shapes are then entered as the starting points ($t = 0$) for our fixture location problem.

For more complicated geometries, key measurement points can be sampled and used to generate initial deformations. Techniques to morph complex parts into non-nominal shapes have been proposed in the literature (Luo *et al.*, 2021).

## 3.3. Reward

The objective of fixture layout design is to minimize shape deformation. Accordingly, the reward is designed in a way that the objective is achieved by maximizing the cumulative rewards, which is

$$r_t = \begin{cases} 0, & t < T \\ \ln \dfrac{1}{d_{\max}}, & t = T. \end{cases} \tag{1}$$

In Equation (1), $d_{\max}$ is the maximum magnitude of deformation over the large-scale sheet part given a specific fixture layout, and $T$ is the overall number of fixtures. In manufacturing practice, the objective of fixture layout design is to ensure that the maximum shape deformation meets the engineering specification. Thus, the reward of RL is designed in inverse proportion to the maximum deformation, such that the maximum deformation can be minimized by maximizing the reward. In addition, the natural logarithm operation is applied to downgrade the magnitude of the reward to facilitate the reward estimation. With the defined reward, the objective function of the optimization problem can be formulated as follows:

$$\max_{\theta, \phi} \mathbb{E}\left[\sum_{i=0}^{T-t} \gamma^i r_{t+i} | s_t\right], \tag{2}$$

where $\theta$ and $\phi$ are trainable parameters in the policy and value functions (introduced in Section 3.6). The discount factor $\gamma \in (0, 1)$ reduces the agent's attention to rewards in the distant future. In our setup, the objective function indicates the expected final maximum deformation (after all fixtures are applied), given the location of the newly applied fixture.

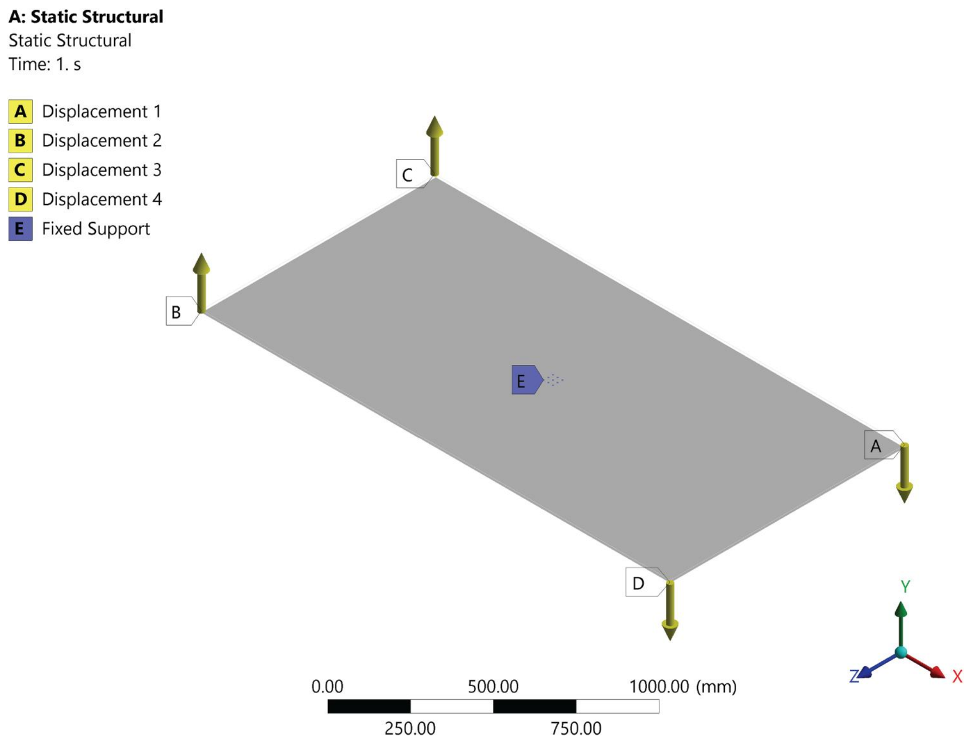

Figure 2. Simulation setup to generate initial deformations for the sheet parts.

### 3.4. Graph-based state representation

The state of our problem is built upon the output from the FEM, given the current fixture layout. With the generated meshes and nodes on the target sheet part and the corresponding boundary conditions (position of fixtures, gravity, etc.), the available output from FEM includes the coordinate of each node (denoted as $(x_i, y_i, z_i)$), the deformation at each node compared with the initial shape (denoted as $(\delta x_i, \delta y_i, \delta z_i)$), and the residual stress at each node (denoted as $(\sigma x_i, \sigma y_i, \sigma z_i)$). The state needs to be properly defined to include the necessary information for the RL agent to select the next action to achieve the design objective. One unique challenge in our problem is that the sheet part might have irregular shapes, which will also change along with different layouts of fixtures. Thus, to better accommodate various shapes and different types of meshes, the state in our problem is represented as a graph built over the nodes, which is denoted as $s = \mathcal{G} \in \mathbb{R}^{N \times k \times M}$. The graph is built by finding the K-Nearest Neighbors (KNN) of each node based on the pairwise Euclidean distance. Suppose we have $N$ nodes in all, and each node has a feature with the dimension of $M$ ($M = 3$ if the directional deformation $(\delta x_i, \delta y_i, \delta z_i)$ is used as the feature). In our case, considering that rectangle meshes are used, the center node in each $3 \times 3$ block is surrounded by eight neighbor nodes. Thus, the number of neighbors of each node is set as eight ($k = 8$) when building the KNN graph. For better illustration, the process to generate the state representation from FEM results is demonstrated in Figure 3.

### 3.5. Engineering-constrained action space

The action in our problem is to select the location of the newly added fixture, given the current state. As the fixture directly contacts and supports the bottom surface of the large-scale sheet part, the nodes on the bottom surface determine the candidate locations of fixtures. More specifically, when a node is selected, its coordinate $(x_i, y_i)$ determines the location of the fixture, and its coordinate $z_i$ determines the height of the fixture, which is demonstrated in Figure 4. For example, suppose there are $N_1$ ($N_1 < N$) nodes on the bottom surface, and all these nodes are candidate locations to place the first fixture.

The action space can be denoted as a matrix $A$ with the shape of $N_1 \times 2$ storing all the candidate locations. In most RL algorithms, the action space remains the same at each time step, which is inapplicable when designing the fixture layout because there are two engineering constraints that need to be considered. First, in an MDP to iteratively add new fixtures, the locations that already have fixtures are infeasible to be selected. Second, when designing the layout of fixtures for a large-scale sheet part, an important rule-of-thumb experience is to ensure the coverage or exploration of the target surface using a given number of fixtures. Thus, placing a fixture close to existing ones is an unfavorable option. To incorporate these two engineering constraints into the action space, instead of keeping $A$ unchanged, our proposed method updates the action space each time after adding a new fixture, which is denoted as $A_t$. More specifically, we maintain a minimum pairwise distance, $2\epsilon$, between the selected locations for fixtures. Thus, for those existing fixtures, their neighbor locations within a distance of $\epsilon$ will be set as infeasible, and the candidate locations for the new fixture will be accordingly reduced to improve the efficiency (demonstrated in Figure 4). The selection of $\epsilon$ is introduced in the experiment setup (Sections 4.1.1 and 4.2.1).

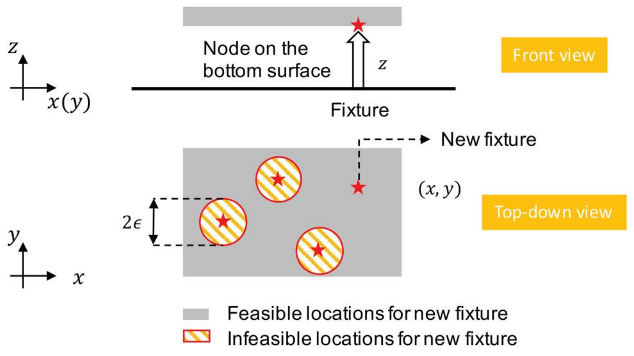
z
x(y)
Node on the bottom surface
z
Fixture
Front view
New fixture
(x, y)
Top-down view
2ϵ
y
x
Feasible locations for new fixture
Infeasible locations for new fixture

Figure 4. Action of selecting fixture location.

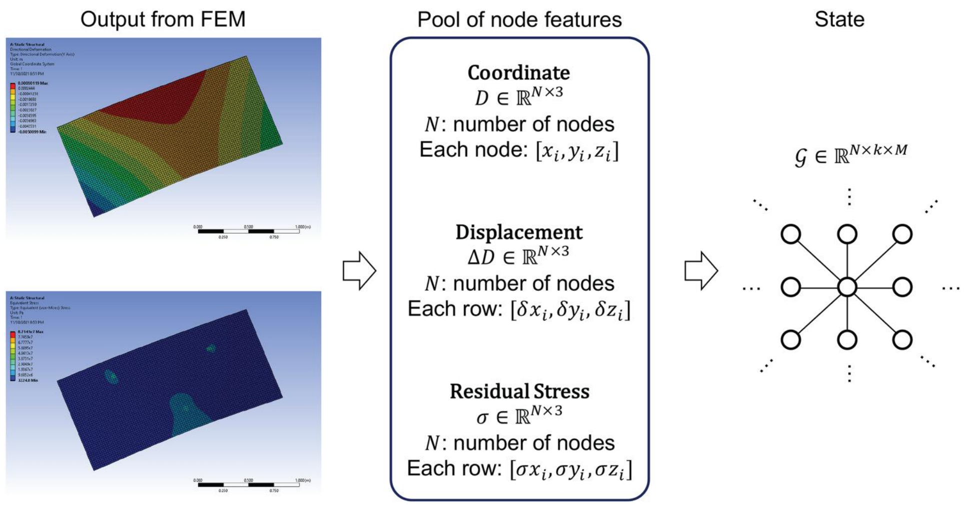
Output from FEM
Pool of node features
State
Coordinate
$D \in \mathbb{R}^{N\times3}$
N: number of nodes
Each node: $[x_i, y_i, z_i]$
Displacement
$\Delta D \in \mathbb{R}^{N\times3}$
N: number of nodes
Each row: $[\delta x_i, \delta y_i, \delta z_i]$
Residual Stress
$\sigma \in \mathbb{R}^{N\times3}$
N: number of nodes
Each row: $[\sigma x_i, \sigma y_i, \sigma z_i]$
$\mathcal{G} \in \mathbb{R}^{N\times k\times M}$

Figure 3. Graph-based state representation from meshes generated by FEM.

## 3.6. Policy and value functions

The policy function $\pi_\theta(a_t|s_t)$ and value function $V_\phi(s_t)$ are approximated by the DNN, which are demonstrated in Figure 5. In our setting, we introduce the idea of AlphaGo Zero (Silver *et al.*, 2017) into the manufacturing and design field. In AlphaGo Zero, the DNNs for the policy and value functions have some shared layers. The advantage of this setting is to reduce the number of parameters ($\theta$ and $\phi$ share some parameters) required in building the model because some common information used for both the policy and value functions are captured by the shared module.

The state generated from the environment is represented as a graph $\mathcal{G} \in \mathbb{R}^{N\times k\times M}$. In the shared module, the Graph Convolutional Layer (GCL) (Wang *et al.*, 2019) is used to extract features from adjacent nodes. The expression of the $l^{\text{th}}$ GCL is given in Equation (3):

$$\mathbf{f}_i^l = \sum_{(i,j)\in\mathcal{G}} h(\Theta_1^l(\mathbf{f}_j^{l-1} - \mathbf{f}_i^{l-1}) + \Theta_2^l \mathbf{f}_i^{l-1}), \tag{3}$$

where $\mathbf{f}_i^l \in \mathbb{R}^{d_l}$ is the feature vector of the $l^{\text{th}}$ node from the $l^{\text{th}}$ GCL; $(i,j)$ represents the edge between node $i$ and $j$; $\Theta_1^l$ and $\Theta_2^l$ are trainable parameters in the $l^{\text{th}}$ GCL; $h(.)$ is the activation function. The expression of GCL indicates that it is designed to extract features from a graph by aggregating and fusing features from adjacent nodes. By stacking multiple GCLs, the information from distant nodes can also be aggregated in the feature extraction. The extracted feature map $F \in \mathbb{R}^{N\times d_L}$ from the shared module is further fed into the policy and value branches to estimate the probability distribution over the current action space $A_t$ (denoted as $\pi_\theta(a_t|s_t)$), and the value of current state $s_t$ (denoted as $V_\phi(s_t)$), respectively. Both the policy and value branches are modeled by a stack of Fully Connected (FC) layers.

## 3.7. Proximal policy optimization

The objective of this article is to train the RL agent to generate the best policy $\pi_\theta(a|s)$ for the fixture layout design such that the expected cumulative reward (defined in Equation (2)) is maximized. The Proximal Policy Optimization (PPO) is selected to fit the parameters in policy and value functions (Schulman *et al.*, 2017). The advantages of PPO make it a good fit to solve our problem, which include: (i) it is an on-policy optimization algorithm, which is designed to improve the policy at each step to converge to the best policy; (ii) it restricts the discrepancy between two adjacent policies, which ensures a stable training process. The loss function used in the PPO is introduced as follows.

The policy and value functions are approximated by the DNNs and have trainable parameters $\theta$ and $\phi$. In PPO, the gradient-based method is used to learn these parameters by optimizing the loss function. For the policy function, the "clip" function is specifically designed to control the update between two adjacent policies, which is given in Equations (4, 5, 6):

$$\begin{aligned} L_1(s_t, a_t, \theta', \theta, \phi) &= \min\left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta'}(a_t|s_t)} G_{\pi_{\theta'}}(s_t, a_t, \phi),\right. \\ &\qquad \left. u\left(e, G_{\pi_{\theta'}}(s_t, a_t, \phi)\right)\right), \end{aligned} \tag{4}$$

$$G_{\pi_{\theta'}}(s_t, a_t, \phi) = \sum_{i=0}^{T-t} \gamma^i r_{t+i} - V_\phi(s_t), \tag{5}$$

$$u\left(e, G_{\pi_{\theta'}}(s_t, a_t, \phi)\right) = \begin{cases} (1+e)G, & G \geq 0, \\ (1-e)G, & G < 0, \end{cases} \tag{6}$$

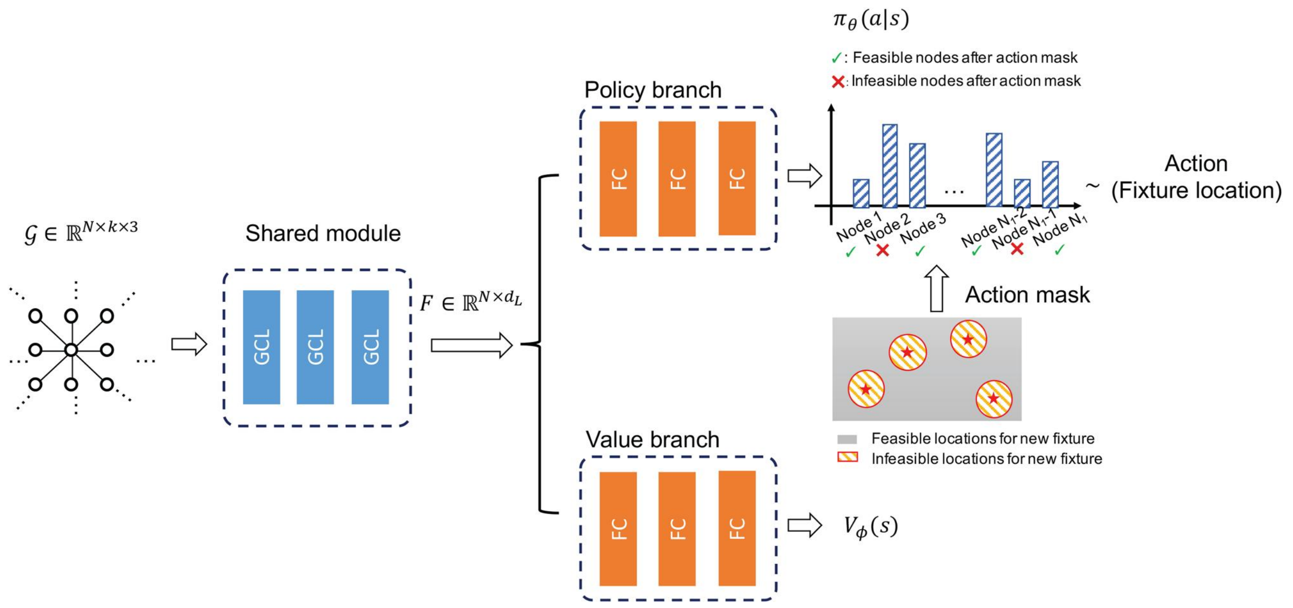
$\mathcal{G} \in \mathbb{R}^{N\times k\times 3}$
Shared module
GCL
GCL
GCL
$F \in \mathbb{R}^{N\times d_L}$
Policy branch
FC
FC
FC
Value branch
FC
FC
FC
$\pi_\theta(a|s)$
✓: Feasible nodes after action mask
✕: Infeasible nodes after action mask
Node 1
Node 2
Node 3
Node $N_f$-2
Node $N_f$-1
Node $N_f$
~ Action (Fixture location)
Action mask
Feasible locations for new fixture
Infeasible locations for new fixture
$V_\phi(s)$

Figure 5. Policy and value functions.

where $G$ is the advantage function describing the difference between the state value and the cumulative future rewards given current action, and $e \in (0, 1)$ defines the limit of changes.

Equation (4) demonstrates that the ratio of probabilities for the same action under two adjacent policies is constrained within $[1-e, 1+e]$. More specifically, if $G \geq 0$, that is, the currently selected action $a_t$ can potentially lead to a better cumulative reward, thus, the probability of selecting $a_t$ (value of $\pi_\theta(a_t|s_t)$) should be increased to receive a better policy. Such an increase is bounded by $1+e$. Similarly, when $G < 0$, the decrease of $\pi_\theta(a_t|s_t)$ will be bounded by $1-e$.

For the value function, the mean squared error between the estimated value of the current state and the actual cumulative reward from the current state is used as the loss function, which is given in Equation (7). The objective of this loss function is to train the value function to estimate the state values accurately

$$L_2(s_t, \phi) = \left( V_\phi(s_t) - \sum_{i=0}^{T-t} \gamma^i r_{t+i} \right)^2 \tag{7}$$

The last term in the PPO loss is the entropy of current policy $\pi_\theta(a|s)$, which is given in Equation (8). A larger value of the entropy loss indicates more randomness in the probability distribution over actions. For example, when the policy tends to converge to a local optimum (e.g., the probability of selecting a sub-optimal action is relatively higher than selecting others), the entropy loss will preserve the ability of the RL agent to explore other actions to improve the current policy further:

$$L_3(a_t, s_t, \theta) = -\pi_\theta(a_t|s_t) \log\left(\pi_\theta(a_t|s_t)\right)^\top \tag{8}$$

The PPO loss is the weighted aggregation of $L_1$, $L_2$, and $L_3$, which is given in Equation (9). The objective of model training is to jointly maximize $L_1$ (improve cumulative reward), minimize $L_2$ (accurately estimate the state value), and maximize $L_3$ (preserve the ability in exploration) by minimizing the PPO loss. Thus, the coefficients $\lambda_1$, $\lambda_2$, and $\lambda_3$ are set as negative, positive, and negative values, respectively

$$\mathcal{L} = \mathbb{E}[\lambda_1 L_1 + \lambda_2 L_2 + \lambda_3 L_3] \tag{9}$$

**Algorithm 1** Pseudo-code of training the proposed Smartfixture.

```
1: Initialize:
   $\theta, \phi$            ▷ initial parameters in policy and value functions.
   $\pi_\theta, V_\phi$      ▷ policy and value functions approximated by DNNs.
   **env**                   ▷ FEA-based environment
2: Training:
3: for $e = 1$ to $E$ do                 ▷ $E$ denote the number of epochs.
4:     $\mathcal{D}_e = \{\}$            ▷ an empty set of sequences in this epoch.
5:     for $p = 1$ to $P$ do             ▷ $P$ denote the number of sequences collected in each epoch.
6:         $D_p = []$                    ▷ an empty list of tuples in this sequence
7:         for $t = 1$ to $T$ do         ▷ $T$ is the number of fixtures in each sequence.
8:             Query state $s_t$ from env.
9:             $a_t =$ Sample from $\pi_{\theta'}(a_t, s_t)$.   ▷ $\theta'$ are parameters from previous epoch.
10:            Iterate **env** by feeding $a_t$ and receive reward $r_t$.
11:            Store tuple $(s_t, a_t, r_t)$ in $D_p$.
12:        end for
13:        Store $D_p$ in $\mathcal{D}_e$.
14:    end for
15:    Evaluate the state values using $V_{\phi'}$ and extend each tuple into $(s_t, a_t, r_t, V_{\phi'}(s_t))$.  ▷ $\phi'$ are parameters from previous epoch.
16:    Calculate the cumulative future rewards and extend each tuple into $(s_t, a_t, r_t, V_{\phi'}(s_t), \sum_{i=0}^{T-t} \gamma^i r_{t+i})$.
17:    Update the parameters $\theta, \phi$ by minimizing the PPO loss with gradient-based optimizer. $\theta', \phi' \leftarrow \text{argmin}_{\theta,\phi} \mathcal{L} = \text{argmin}_{\theta,\phi} \mathbb{E}_{\mathcal{D}_e}[\lambda_1 L_1 + \lambda_2 L_2 + \lambda_3 L_3]$
18: end for
```

In summary, the process of training the proposed SmartFixture using PPO is summarized in Algorithm 1. The step-wise demonstration to collect the trajectory of a single episode is shown in Figure 6. In the context of our problem, it indicates generating the sequence of tuples $(s_t, a_t, r_t), t = 1, \ldots, T$, by placing fixtures sequentially.

## 4. Case study

In this section, we will introduce the details of the experiment setup and compare the performances between the proposed method and benchmark methods. Section 4.1 introduces the experiment results on the sheet metal and composite part without initial shape deformation, which is used as a proof-of-concept to demonstrate the capability of the RL-based method. Section 4.2 demonstrates the experiment results on sheet metal and composite parts with different initial deformations, which is designed to show the generalizability of the proposed SmartFixture.

### 4.1. Case-1: Fixture layout design for ideal parts

#### 4.1.1. Experiment setup

The first experiment is used as a proof-of-concept to demonstrate the effectiveness of the proposed method in generating the optimal fixture layout to minimize shape deformations for sheet parts made of metal or composite. Sheet metal or composite parts are widely used in the manufacturing process of many products, such as ship hulls, marine structures, automobiles, aircraft, etc. The shape control of sheet parts has a large impact on the quality of final products. In this case, only the deformation caused by

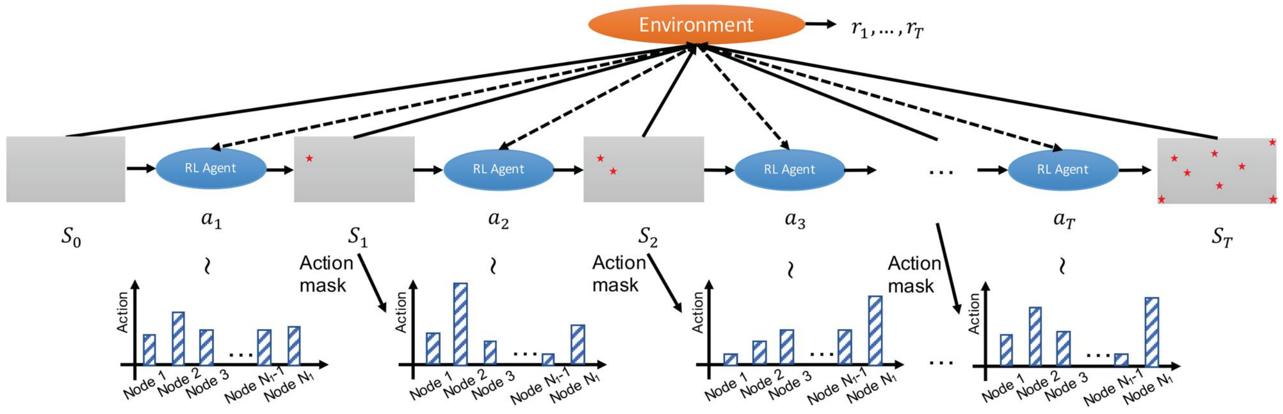

**Figure 6.** Step-wise demonstration of collecting a sequence of tuples.

gravity is considered, and the sheet part, either metal or composite, is assumed to have negligible initial deformations. The detailed simulation setup can be referred to in Section 3.2. The two types of sheet parts share the same length and width (2 m × 1 m). It indicates these two types of parts have the same surface size to which fixtures can be applied. Different RL agents are trained under the same setting to design fixture layouts for the sheet part made of different materials. The state $s_t$ here is defined to include the coordinate of each node and the deformation at each node. Every time the location of the newly added fixture is selected by the RL agent, FEA will update the state by simulating deformations over all nodes, given the current fixture layout. It is worth noting that shape deformation is not available when the number of fixtures is less than three (at least three points are needed to determine a plane). Thus, the shape deformation included in $s_t$ is assumed to be uniformly zero when $t < 3$.

There are over a thousand candidate locations for the layout of eight fixtures in our problem. The minimum pairwise distance $\epsilon$ (introduced in Section 3.5) determines the design space. It is worth noting that the design space with a larger $\epsilon$ is the subset of the design space with a smaller one, which is a trade-off between shrinking the design space with engineering insights for better efficiency and enabling the exploration of different layouts. Considering the target surface to place eight fixtures has the shape of $2\ \text{m} \times 1\ \text{m}$, thus, if they are placed evenly in two rows along the 2 m edge, the average pairwise distance among them can be roughly estimated as 0.5 m. In other words, the minimum pairwise distance is "roughly" at most 0.5 m for the deployment of eight fixtures over the target surface. To allow a larger design space, we search the value of $\epsilon$ within the range [0.4, 0.45] m under the same training time and receive the best results with $\epsilon = 0.45\ \text{m}$.

To demonstrate the advantage of the proposed method, we select two types of symmetric layouts (denoted as "Sym 1" and "Sym 2" when demonstrating the results) and the layout generated from the latest heuristic optimization method (Du *et al.*, 2021) as benchmarks. In practice, the key rule of thumb for designing the fixture layout is to spread the programmable fixtures over the target surface evenly. Thus, the two symmetric layouts are designed according to the shape of the target surface of sheet parts (kept the same for both metal and composite parts). Both symmetric layouts placed four fixtures at the four corners of the sheet part. One layout placed the remaining four fixtures in the shape of a smaller rectangle. The other layout placed one fixture at the center of the sheet part and placed the remaining three in the shape of an equilateral triangle. The heuristic optimization method is built upon simulated annealing (denoted as "SA" or "baseline" when demonstrating the results) following the parameters used in Du *et al.* (2021), which might converge to different results when repeated multiple times. Thus, for each type of sheet part (in metal or composite), we repeat the SA method five times and report the average and variance of its performance. To fully demonstrate the performance of the SA method, each time it is repeated, it will be terminated after searching 1500 different layouts. In other words, given the SA method is repeated five times for each type of sheet part, there will be a total of 1500 × 5 different layouts explored by this method for each type of sheet part, which requires far more simulations than the proposed RL method.

### 4.1.2. Experiment results

The results for the fixture layout optimization of the sheet metal or composite parts are demonstrated in Figure 7. For the SA method, we select the best layout (with the smallest maximum magnitude of deformation) received from five replications for the experiment. The maximum magnitude of deformation is calculated to compare the performance of different layouts. From the results, we can observe: (i) the fixture layout generated by the proposed method also includes positions close to corners (i.e., fixtures #3, 5, 7, 8 for sheet metal part and fixtures #1, 2, 3, 4 for sheet composite part), which reveals that the proposed method successfully learned some experience consistent to our intuition; (ii) our proposed method also shows "novelty" in fixture layout design to receive a smaller overall deformation by partially sacrificing the symmetric rule that a human designer commonly follows; (iii) compared with the heuristic-based optimization, the learning-based method captured some useful rules when designing the fixture layout (i.e., spread the fixture over target surface).

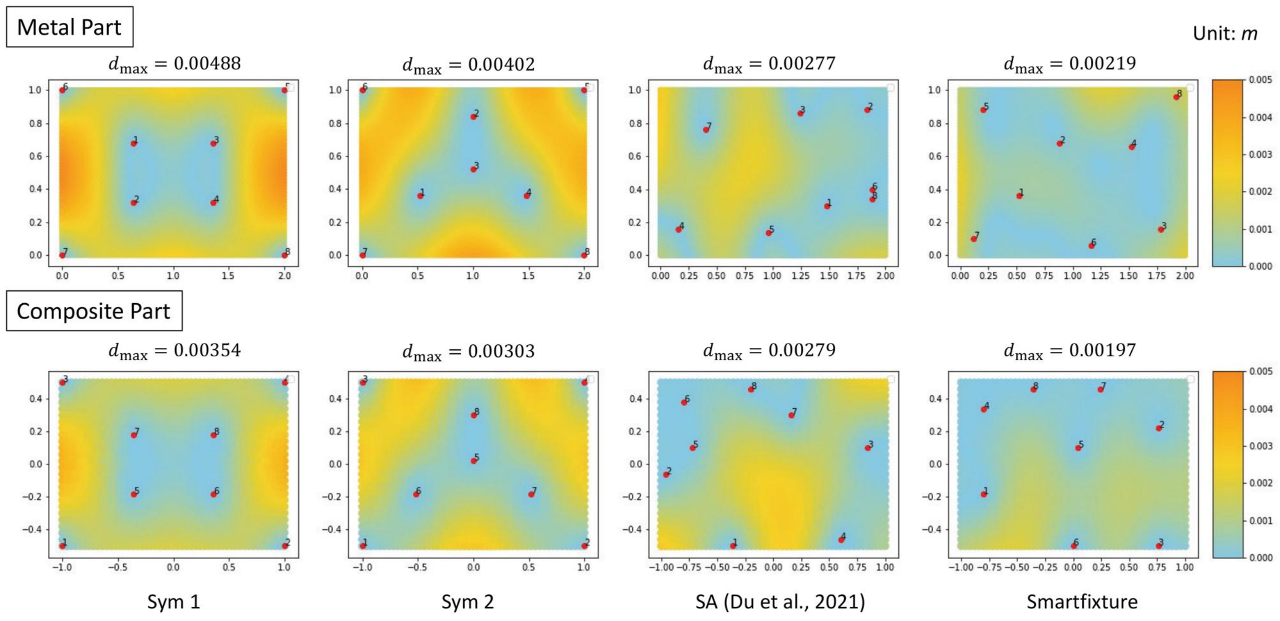

**Figure 7.** Fixture layouts for ideal parts.

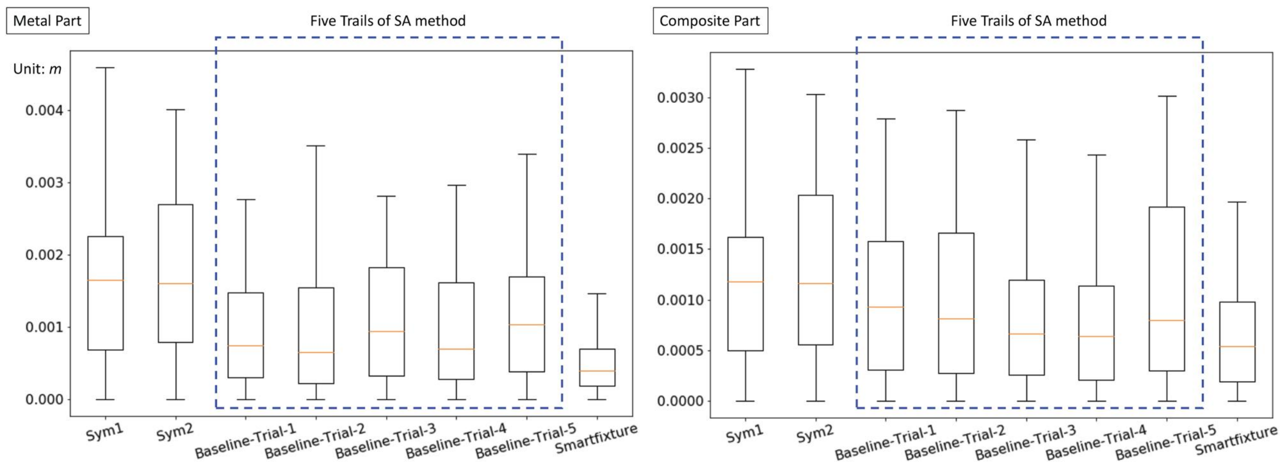

**Figure 8.** Distribution of the magnitude of deformation (without initial deformations). **Note**: Five trials represent five replications of the SA method.

To better quantitatively evaluate the performance of our proposed method, we also demonstrate the magnitude of deformation over the target surface in boxplot (in Figure 8), from which we can conclude that for sheet parts made of steel or composite: (i) the interquartile range of the magnitude of deformation (deformations of half of the nodes) achieved by the proposed method is less than 0.001 m for the sheet part, which is much better than the benchmark methods; (ii) the variation of deformations are also improved by the proposed method; (iii) when repeating the SA method for five times (highlighted in blue dashed box), the performance has a relatively large variance over different trials. For the sheet metal part, the maximum shape deformations over five trials of the SA method receive an average of 0.00324 m and a standard deviation of 0.0005 m. For the sheet composite part, the maximum shape deformations over five trials of the SA method receive an average of 0.00289 m and a standard deviation of 0.0001 m.

In summary, the results demonstrate that our proposed method successfully learns general principles in fixture layout design (i.e., placing fixtures close to corners to improve space-filling) and shows novelty in the overall fixture layout. It consistently outperforms the benchmark methods on both ideal sheet metal and composite parts and receives the smallest shape deformation after applying the designed fixtures.

## 4.2. Case-2: Fixture layout design for parts with initial deformations

### 4.2.1. Experiment setup

In the manufacturing process involving sheet metal and composite parts, every single part inevitably has some initial deformations, which will be another factor influencing the quality of the final product. The initial deformations might vary among different sheet parts, which require customized designs of fixture layouts to accommodate different scenarios. In our experiment setting, different initial deformations can be treated as different initial conditions for the environment. The challenge is that the same action (location of the fixture) from the RL agent can lead to different rewards under different scenarios. The heterogeneity of the environment requires the proposed RL algorithm to understand the general principles of fixture design that can be generalized to customize the fixture layout design for sheet parts with different initial deformations.

The FEA simulation shown in Figure 2 can generate sheet parts made of metal or composite with different initial deformations by specifying the displacements caused by each force. To better represent all possible initial deformations, an optimal space-filling design (Joseph, 2016) is conducted to determine the set of combinations of displacements for these four forces. For each type of sheet part, a total of 25 different initial deformations are generated. Of these, 20 are randomly selected to train the proposed SmartFixture, and the remaining five are used to test the model performance. Similarly, the FEA simulation needs locations of at least three fixtures to generate the deformation caused by gravity. Thus, only the initial deformation at each node is available when selecting the positions of the first three fixtures. The value of $\epsilon$ is also searched within [0.4, 0.45] m, and the best results are received when $\epsilon = 0.42$ m. Similarly, different RL agents are trained for sheet parts with different materials on their corresponding training dataset. During the training, for a specific initial deformation, it will interact with the environment to collect the tuples of action, state, and reward as the training sample, which will be repeated by resetting the environment to its initial condition until enough training samples are generated. We expose the proposed method with 20 different initial deformations in the training phase and set aside five unseen scenarios to test the generalizability of learned principles. Here, we also select the same symmetric fixture layouts and the SA method as benchmarks. For each testing scenario, the SA method also repeats five times, and each repetition will be terminated after 1500 different layouts are generated.

### 4.2.2. Experiment results

The experiment is designed to train the proposed SmartFixture to generate fixture layouts for the sheet metal and composite parts with different initial deformations, respectively. The results of five unseen scenarios for sheet metal and composite parts are visualized in Figure 9 and Figure 10, respectively. In both figures, the first row shows the sheet parts with different initial deformations, and the rest of the rows show the corresponding fixture layouts generated by different methods. The results of the SA method are also the best ones over five trials on each scenario. We found that both the SA method and our proposed SmartFixture outperformed the symmetric fixture layout in all test scenarios. In addition, SmartFixture is better than, or comparable to, the SA method in most scenarios, with the exception of the design point (dp) #14 ("dp14") for the sheet metal part and "dp17" for the sheet composite part. The reason can be summarized as (i) the SmartFixture has no exposure to

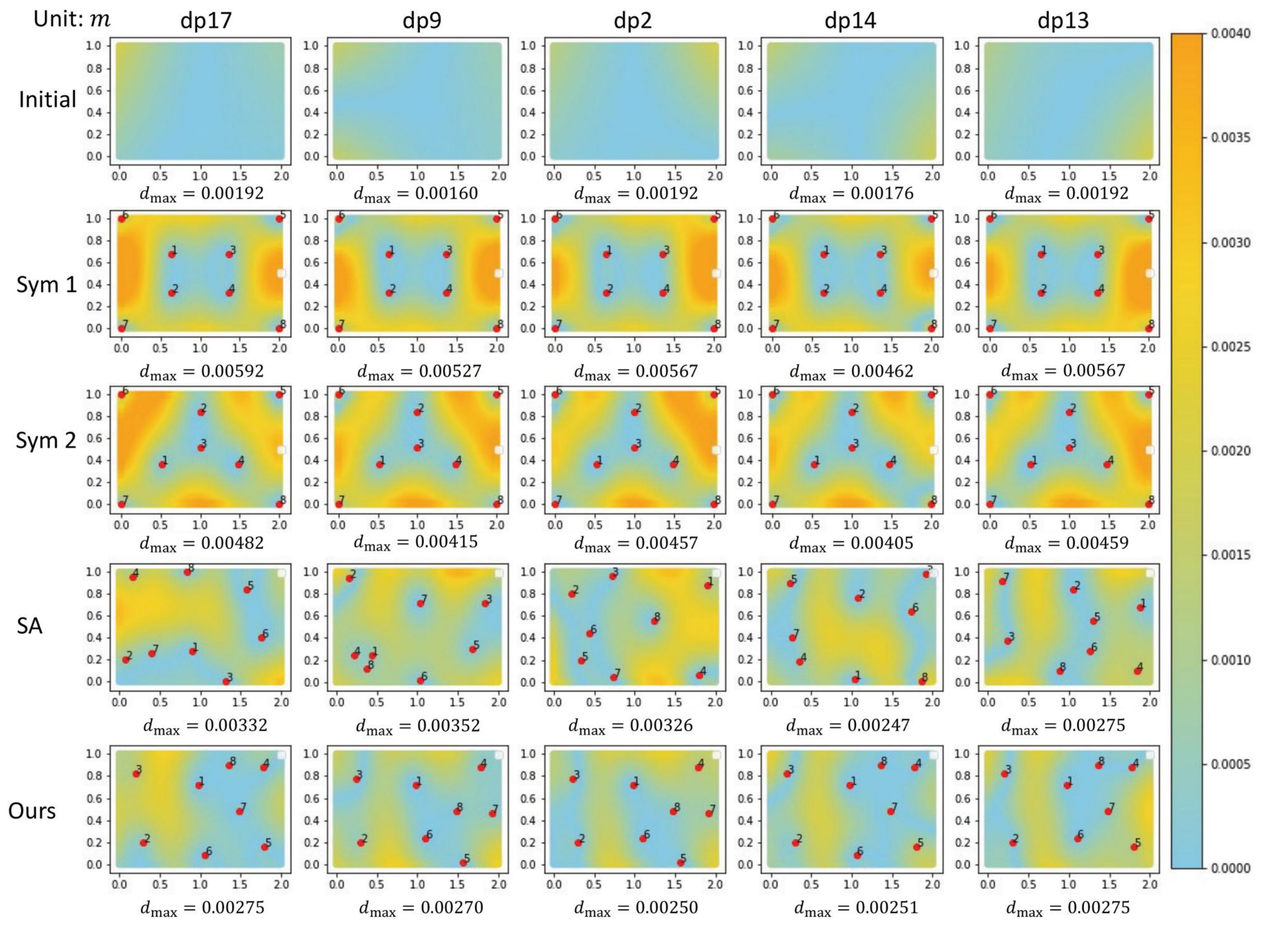

Figure 9. Fixture layouts for sheet metal parts with initial deformations.

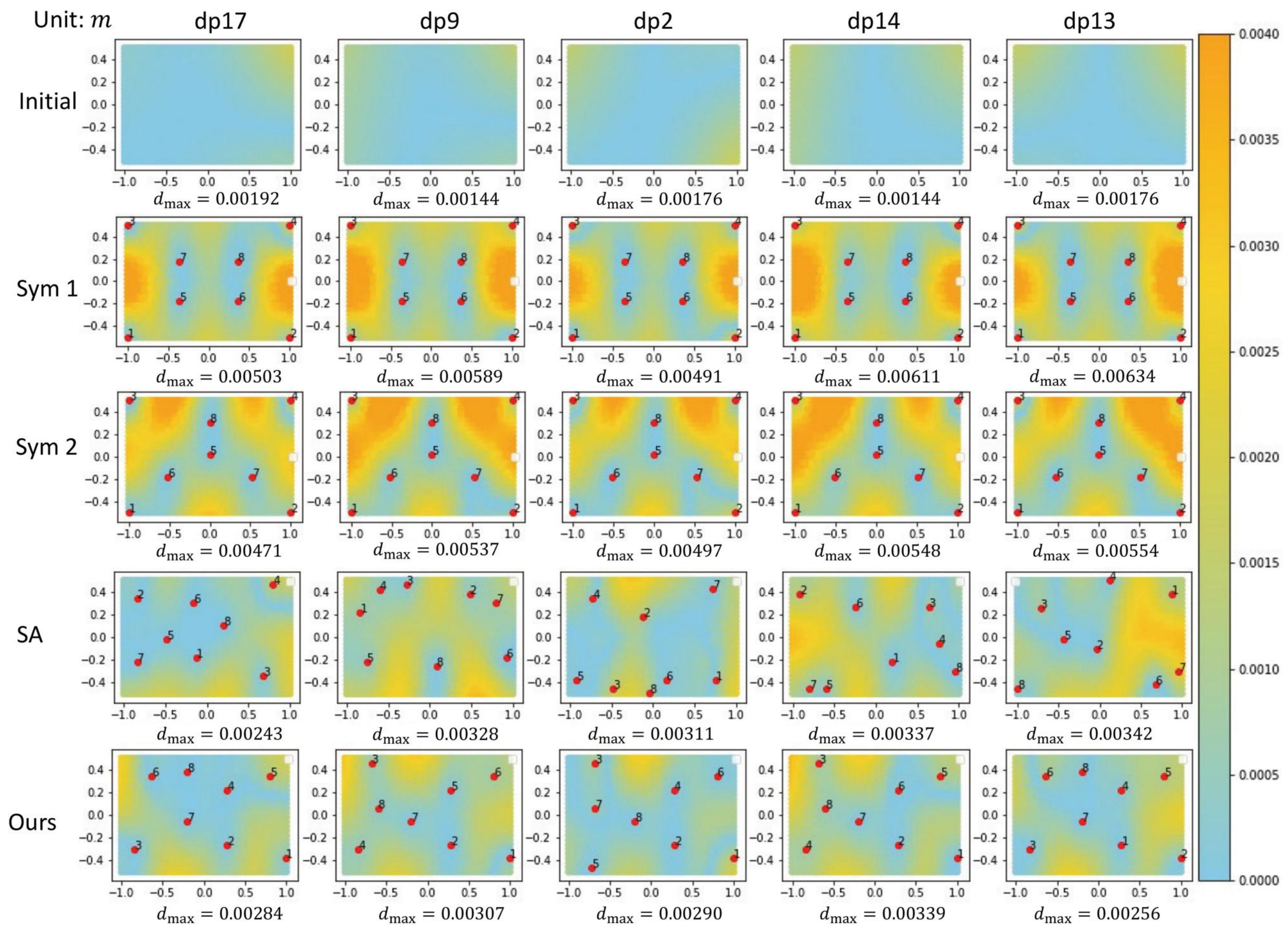

**Figure 10.** Fixture layouts for sheet composite parts with initial deformations.

**Table 2.** Maximum magnitude of deformation over different test scenarios (m).

| | *Metal* | | | *Composite* | | |
|---|---|---|---|---|---|---|
| *Method* | *Mean* | *Range* | *Std.* | *Mean* | *Range* | *Std.* |
| Sym1 | 0.00543 | 0.00130 | 0.00045 | 0.00566 | 0.00143 | 0.00058 |
| Sym2 | 0.00443 | 0.00077 | 0.00029 | 0.00521 | 0.00083 | 0.00032 |
| SA (Du *et al.*, 2021) | 0.00306 | 0.00105 | 0.00039 | 0.00312 | 0.00099 | 0.00036 |
| SmartFixture | 0.00264 | 0.00025 | 0.00011 | 0.00295 | 0.00083 | 0.00027 |

these five testing scenarios in training and does not repeat the optimization process in the testing phase, whereas the SA method repeats the heuristic optimization process on each scenario and selects the best result; (ii) when applying in the testing phase, the SmartFixture can efficiently generate the fixture layout at once whereas the result of SA method is selected from $1500 \times 5$ layouts in our setting.

The summary of the maximum magnitude of deformations over all the test scenarios is given in Table 2. From the results, we can conclude that our proposed method receives consistently outstanding performance with the smallest mean and variation over different unseen scenarios, which indicates an outstanding generalization ability.

The deformations over the entire target surface on unseen scenarios are also demonstrated in the boxplot (Figure 11), in which the left column demonstrates the five unseen scenarios for the sheet metal part, and the right column demonstrates the five unseen scenarios for the sheet composite part. For the SA method, we demonstrate the deformation distribution from all five trials, which are shown in blue dashed boxes. When comparing the results among different trials of the SA method, we can clearly see a large variation. To better compare the results from our proposed SmartFixture and SA method, we marked the upper bound and median of the boxplot in green and red dashed lines, respectively. For sheet parts made of both types of materials, we can conclude that the SmartFixture consistently receives the smallest maximum and median deformations when compared with all benchmark methods in most cases. In addition, all the median lines in five test scenarios are below the 0.001 m, which means half of the measurement points on the target surface have deformation less than 0.001 m.

### 4.2.3. Step-wise demonstration on learned principles

We further use the sheet metal part as an example to demonstrate the step-wise process of fixture layout generation in Figure 12 to show the principles of selecting fixture positions learned by our proposed method. New fixtures are continuously placed until the total number of fixtures reaches the limit. The magnitude of deformation jointly caused by initial deformation and gravity is visualized. We can observe that:

1. Although the initial deformations are different when the number of fixtures is small (i.e., $n \leq 4$), the difference among test scenarios is not significant because the

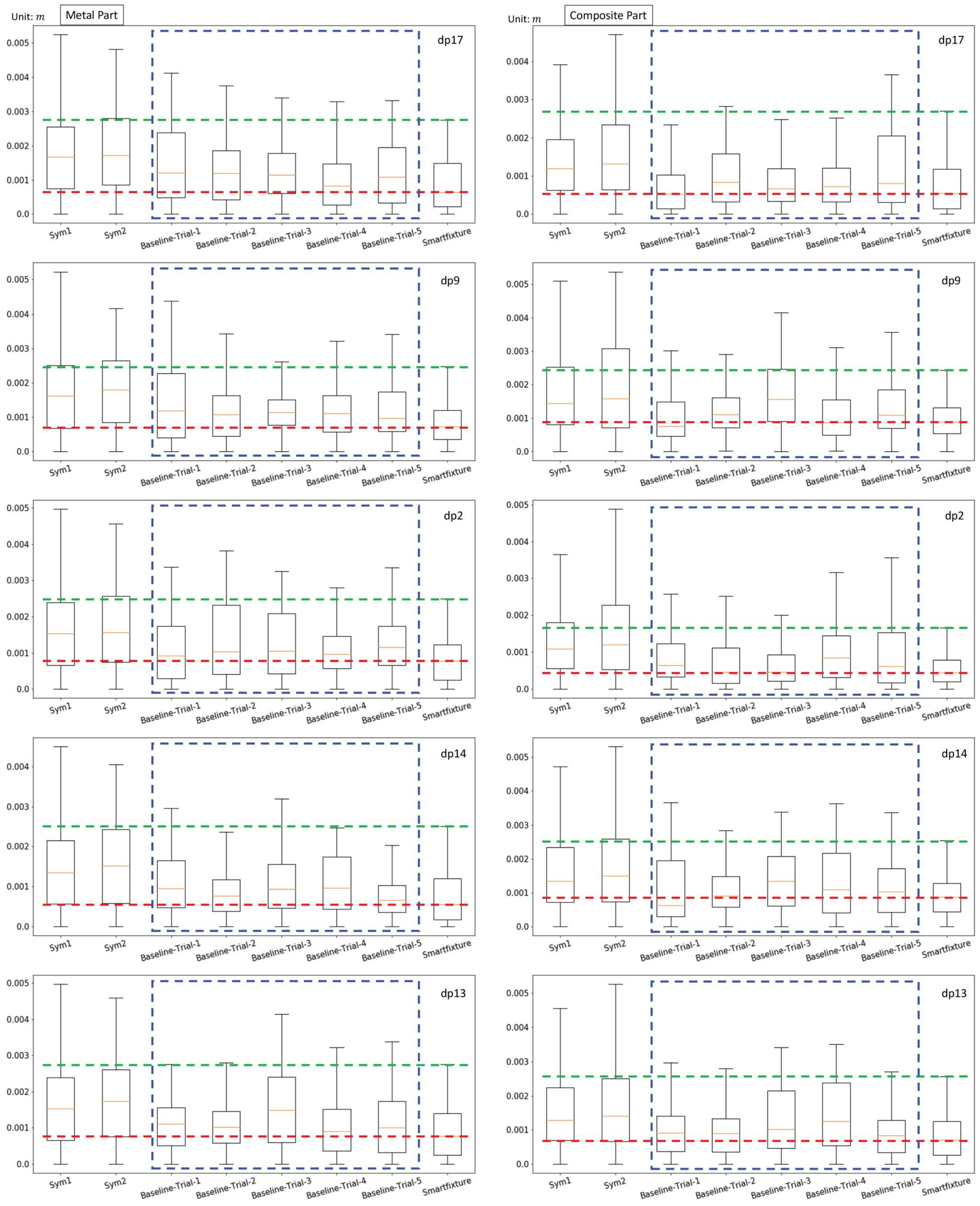

**Figure 11.** Distribution of the magnitude of deformation (with initial deformations).

effects of gravity dominate the deformations. Thus, the locations of the first four fixtures are the same for different test cases.

2. Starting from $n = 4$, the magnitudes of deformations start to show different distributions over the sheet metal part (highlighted in white dashed boxes), and such different distributions lead to different locations of newly added fixtures. For example, when $n = 4$, the different deformation distributions in the highlighted regions lead to different locations of the fifth fixture. Similar observations also existed when determining the locations of the remaining three fixtures. The step-wise demonstration reveals that our proposed method learned general principles in determining the fixture locations, which can be transferred to unknown

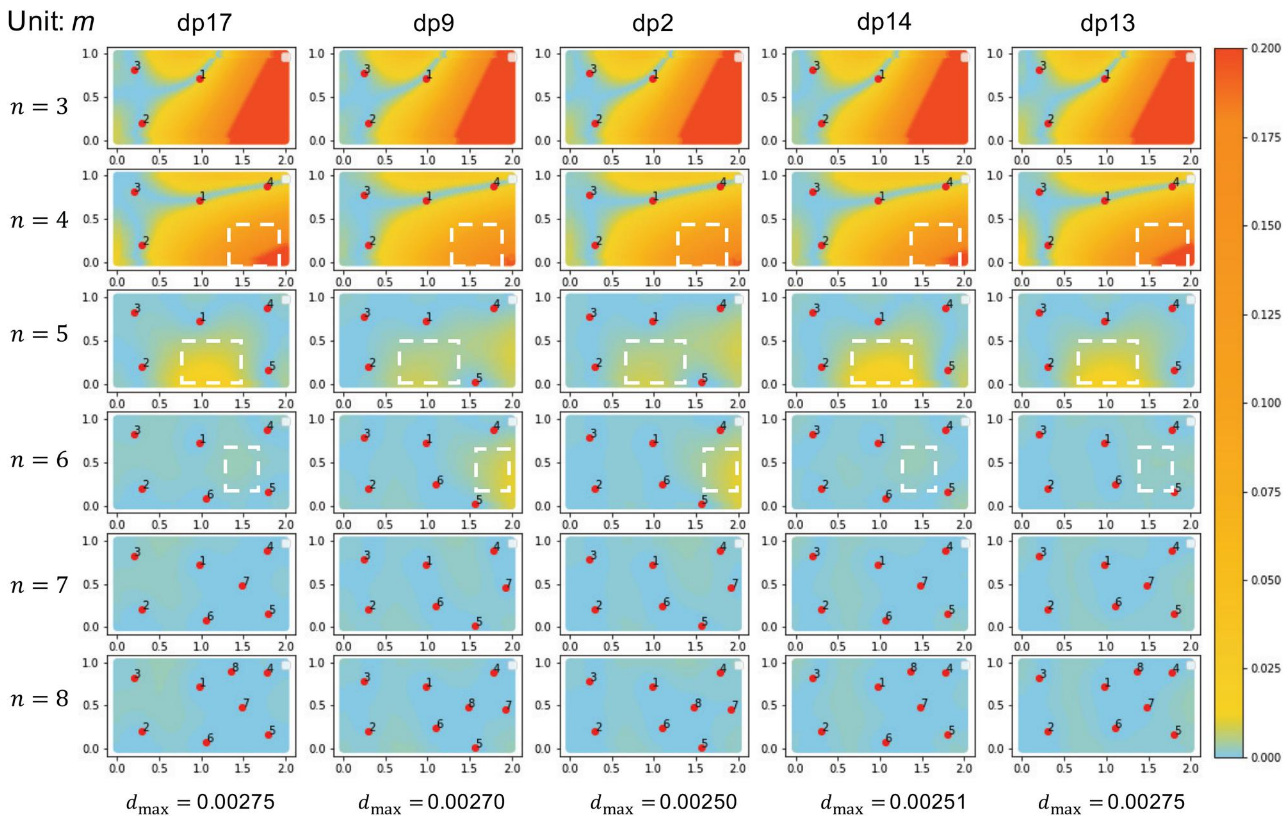

**Figure 12.** Step-wise demonstration when generating fixtures for parts with initial deformations.

scenarios. Such findings also indicate that the RL agent started to master the "game" of fixture layout design via the interaction with the FEA-based "environment", just like how it masters the Go game, which also has the scalability to handle more complex design problems in the manufacturing process.

### 4.2.4. Discussion on computational complexity

In the training process, the bottleneck of computational cost is the interaction with the FEA simulation. In the current setup of simulation, it takes approximately 1 second to run the simulation once. Running the simulation once is defined as feeding the generated action (layout of fixtures) into FEA and calculating the state and reward (deformation of sheet part). As the RL algorithm is designed to place a total of eight fixtures sequentially, six runs of simulation are needed to collect one full trajectory of action, state, and reward (the simulation starts to run after at least three fixtures are placed). Therefore, in the training phase, if we need to collect trajectories of 1000 different fixture layouts, there will be 6000 simulations to run, which takes approximately 100 minutes. When we consider different initial deformations, we need to further increase the number of different layouts explored in the training phase to learn the general principle that accommodates different scenarios. The real training of the RL agent (updating model parameters and policy after collecting the trajectories) is actually very efficient when deployed the PPO algorithm on the GPU. We usually are not worried about the time cost of running simulations as there are multiple solutions to mitigate or resolve this issue. For example, multiple simulations can run in parallel to simultaneously collect trajectories for different layouts, which will significantly improve the efficiency. In addition, the idea of multi-fidelity simulation is popular. It can efficiently generate simulation results by hybridly using high-fidelity FEA or low-fidelity surrogate models. It is also worth noting that, after training, the RL algorithm demonstrated outstanding generalizability to be directly applied to unseen scenarios without further training, which has been shown in the results. However, the classic optimization method does not have this capability, which means that the optimization has to be repeated from scratch whenever a new scenario arises. It is insufficient in current manufacturing practice.

In summary, when the initial deformations of sheet parts vary, the MO-based methods need to repeat the optimization process from scratch, whereas our proposed method demonstrates the generalizability of transferring learned knowledge and experience to unseen scenarios. The results indicate that after training, the proposed method can efficiently customize the fixture layout design for sheet parts according to their initial deformations to achieve promising performance in reducing shape deformation.

## 5. Conclusion

Manufacturing process design is an emerging topic that ensures product quality in advanced manufacturing. In this article, we propose a learning-based optimization framework, SmartFixture, to directly interact with the RL agent with FEA-based simulation to optimize the fixture layout design for large-scale sheet parts. More specifically, the FEA-based simulation is built to generate deformations of the sheet metal and composite parts given a specific fixture layout. The RL agent will iteratively read outputs from FEA and generate the location of the newly added fixture to update the boundary conditions in FEA. The case studies demonstrate that (i) the proposed SmartFixture consistently outperforms the benchmark methods in designing the fixture layout to receive a smaller shape deformation; (ii) it also has a good generalization ability to design fixture layouts for sheet parts with different initial deformations. The methodology contributions of the proposed method include: (i) it builds a highly scalable framework to enable the interaction directly between learning-based optimization and FEA-based simulation, which can be extended to optimize various types of manufacturing processes; (ii) the learning-based optimization demonstrated outstanding generalizability to transfer the learned knowledge to unseen scenarios. It provided innovative solutions and received much better performance compared with the traditional symmetric design of fixture layout, which opened a new direction in the manufacturing design; (iii) it significantly extended the scale and complexity of the design space that can be explored and optimized compared with the heuristic and nonlinear optimization techniques, which aligns the practical needs in the design of manufacturing processes.

For future work in this study, we plan to investigate two directions. First, we plan to extend the simulation from three aspects: (i) include diversified sheet parts with arbitrary shapes that are commonly used in the industry, (ii) consider the multi-stage assembly process, and (iii) generalize the controllable variables to consider the impact of both locations and forces of fixture or actuators on the product shape deviation. Second, we plan to further improve the developed learning-based optimization method to (i) consider the cascade dependency among multi-stage assembly processes to reduce the propagation and accumulation of shape deviations, (ii) improve the sample efficiency and reduce the number of required simulations for the assembly process, and (iii) systematically quantify the uncertainties in the manufacturing process and evaluate how it will impact the performance of the proposed method.

## Acknowledgments

The authors thank Dr. Zhenyu (James) Kong for the insightful discussions and constructive suggestions.

## Funding

Dr. Xiaowei Yue was supported by the Beijing Natural Science Foundation under ID 3244032. The authors also acknowledge the funding support from the U.S. National Science Foundation under Award CMMI-2035038 and the Grainger Frontiers of Engineering Grant Award from the U.S. National Academy of Engineering.

## Notes on contributors

*Dr. Yinan Wang* serves as the tenure-track assistant professor in the Department of Industrial and Systems Engineering at Rensselaer Polytechnic Institute. He received a PhD degree in the Grado Department of Industrial and Systems Engineering at Virginia Tech. His research interest lies in developing systematic engineering-driven machine learning methodologies with applications to advanced manufacturing. He is a recipient of FTC Early Career Award, seven best paper awards (e.g., IISE Transaction Best Paper Award), and two best dissertation awards. He also won the Faculty Achievement Award from RPI.

*Dr. Tim Lutz* is a Clinical Assistant Professor in the Darla Moore School of Business at the University of South Carolina. He received the PhD degree in the Grado Department of Industrial and Systems Engineering at Virginia Tech. His research interest lies in computational simulation and data analytics for aerospace manufacturing.

*Dr. Xiaowei Yue* is an associate professor with the Department of Industrial Engineering, Tsinghua University. Prior to that, he was an assistant professor with the Grado Department of Industrial and Systems Engineering, Virginia Tech, Blacksburg, USA. His research interests include machine learning for advanced manufacturing. He is a recipient of the IISE Hamed K. Eldin Outstanding Early Career IE in Academia Award, the SME Outstanding Young Manufacturing Engineer Award, the IISE Manufacturing and Design Young Investigator Award. He received the Grainger Frontiers of Engineering Grant Award from the U.S. National Academy of Engineering (NAE). He serves as an associate editor for *IISE Transactions*, *IEEE TASE,* and *IEEE TNNLS.* He is selected to be an Editorial Board Member for *PNAS Nexus*, an open-access journal of the U.S. National Academy of Sciences (NAS). He is a senior member of IISE, and is selected to be the president-elect of IISE QCRE division.

*Prof. Juan Du* holds a PhD degree in management science and engineering from the Department of Industrial Engineering & Management at Peking University and a BEng in Composite Materials and Engineering from Honors School of Harbin Institute of Technology. She had been a Visiting/Join PhD Student at H. Milton Stewart School of Industrial & Systems Engineering of Georgia Tech from 2017.9-2019.9. She worked as an assistant professor at the Department of Industrial Engineering & Management at Shanghai Jiao Tong University, Shanghai, China before joining the Smart Manufacturing Thrust, Systems Hub, The Hong Kong University of Science and Technology (Guangzhou) as an assistant professor in 2021. Her research interest focuses on data analytics and system informatics and control for performance improvements in complex systems.

## ORCID

Yinan Wang http://orcid.org/0000-0002-4079-1658
Xiaowei Yue http://orcid.org/0000-0001-6019-0940
Juan Du http://orcid.org/0000-0002-6018-2972

## Data and code availability

The data and code of this work will be available via the link: https://github.com/wyn430/SmartFixture.

## References

AlBahar, A., Kim, I., Lutz, O.T., Wang, X. and Yue, X. (2024) Stress-aware optimal placement of actuators for high precision quality management in composite aircraft assembly. *IEEE Transactions on Automation Science and Engineering*, Early Access. https://doi.org/10.1109/TASE.2023.3347357

Cai, W., Hu, S.J. and Yuan, J.X. (1996) Deformable sheet metal fixturing: Principles, algorithms, and simulations. *Journal of Manufacturing Science and Engineering*, **118**(3), 318–324.

Camelio, J.A., Hu, S.J. and Ceglarek, D. (2004) Impact of fixture design on sheet metal assembly variation. *Journal of Manufacturing Systems*, **23**(3), 182–193.

Canzini, E., Auledas-Noguera, M., Pope, S. and Tiwari, A. (2024) Decision making for multi-robot fixture planning using multi-agent reinforcement learning. *IEEE Transactions on Automation Science and Engineering*, Early Access. https://doi.org/10.1109/TASE.2024.3424677

Chung, J., Shen, B., Law, A.C.C. and Kong, Z.J. (2022) Reinforcement learning-based defect mitigation for quality assurance of additive manufacturing. *Journal of Manufacturing Systems*, **65**, 822–835.

Du, J., Liu, C., Liu, J., Zhang, Y. and Shi, J. (2021) Optimal design of fixture layout for compliant part with application in ship curved panel assembly. *Journal of Manufacturing Science and Engineering*, **143**(6), 061007.

Du, J., Yue, X., Hunt, J.H. and Shi, J. (2019) Optimal placement of actuators via sparse learning for composite fuselage shape control. *Journal of Manufacturing Science and Engineering*, **141**(10), 101004.

He, J., Tang, M., Hu, C., Tanaka, S., Wang, K., Wen, X.-H. and Nasir, Y. (2022) Deep reinforcement learning for generalizable field development optimization. *SPE Journal*, **27**(01), 226–245.

He, K., Zhang, X., Ren, S. and Sun, J. (2016) Deep residual learning for image recognition, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE Press, Piscataway, NJ, pp. 770–778.

Howard, R.A. (1960) *Dynamic Programming and Markov Processes*. John Wiley and Sons, New York.

Huang, W., Kong, Z. and Chennamaraju, A. (2010) Robust design for fixture layout in multistation assembly systems using sequential space filling methods. *Journal of Computing and Information Science in Engineering*, **10**(4), 041001.

Joseph, V.R. (2016) Space-filling designs for computer experiments: A review. *Quality Engineering*, **28**(1), 28–35.

Kaszynski, A. (2020) pyansys: Python interface to MAPDL and associated binary and ASCII Files (0.43.2). Zenodo.

Kim, P. and Ding, Y. (2004) Optimal design of fixture layout in multistation assembly processes. *IEEE Transactions on Automation Science and Engineering*, **1**(2), 133–145.

Kiran, B.R., Sobh, I., Talpaert, V., Mannion, P., Al Sallab, A.A., Yogamani, S. and Pérez, P. (2021) Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, **23**(6), 4909–4926.

Kulankara, K., Satyanarayana, S. and Melkote, S.N. (2001) Iterative fixture layout and clamping force optimization using the genetic algorithm. *Journal of Manufacturing Science and Engineering*, **124**(1), 119–125.

Lee, C., Wu, J., Wang, W. and Yue, X. (2022) Neural network Gaussian process considering input uncertainty for composite structure assembly. *IEEE/ASME Transactions on Mechatronics*, **27**(3), 1267–1277.

Low, D.W.W., Neo, D.W.K. and Kumar, A.S. (2020) A study on automatic fixture design using reinforcement learning. *The International Journal of Advanced Manufacturing Technology*, **107**(5), 2303–2311.

Luo, C., Franciosa, P., Ceglarek, D., Ni, Z. and Mo, Z. (2021) Early stage variation simulation and visualization of compliant part based on parametric space envelope. *IEEE Transactions on Automation Science and Engineering*, **18**(3), 1505–1515.

Lutz, T., Yue, X. and Camelio, J. (2022) Towards a digital twin: Simulation and residual stress analysis in aerospace composite structures assembly, in *Proceedings of the ASME 2022 17th International Manufacturing Science and Engineering Conference*. Volume 2: Manufacturing Processes; Manufacturing Systems. West Lafayette, Indiana, USA. June 27–July 1, 2022. V002T05A036. ASME.

Madenci, E. and Guven, I. (2015) *The Finite Element Method and Applications in Engineering using ANSYS®*, Springer, New York, NY.

Menassa, R.J. and DeVries, W.R. (1991) Optimization methods applied to selecting support positions in fixture design. *Journal of Engineering for Industry*, **113**(4), 412–418.

Mirhoseini, A., Goldie, A., Yazgan, M., Jiang, J.W., Songhori, E., Wang, S., Lee, Y.-J., Johnson, E., Pathak, O., Nazi, A. *et al* (2021) A graph placement methodology for fast chip design. *Nature*, **594**(7862), 207–212.

Mou, S., Biehler, M., Yue, X., Hunt, J.H. and Shi, J. (2023) Spac: Sparse sensor placement-based adaptive control for high precision fuselage assembly. *IISE Transactions*, **55**(11), 1133–1143.

Nasir, Y., He, J., Hu, C., Tanaka, S., Wang, K. and Wen, X. (2021) Deep reinforcement learning for constrained field development optimization in subsurface two-phase flow. *Frontiers in Applied Mathematics and Statistics*, **7**, 689934.

Parks, O.F. (2020) The future of composites for marine applications. PhD thesis, University of Bristol, Bristol, UK.

Pavlović, A., Sintoni, D., Minak, G. and Fragassa, C. (2020) On the modal behaviour of ultralight composite sandwich automotive panels. *Composite Structures*, **248**, 112523.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A. and Klimov, O. (2017) Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Shroff, S., Acar, E. and Kassapoglou, C. (2017) Design, analysis, fabrication, and testing of composite grid-stiffened panels for aircraft structures. *Thin-Walled Structures*, **119**, 235–246.

Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., *et al* (2016) Mastering the game of Go with deep neural networks and tree search. *Nature*, **529**(7587), 484–489.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T., Simonyan, K. and Hassabis, D. (2018) A general reinforcement learning algorithm that masters Chess, Shogi, and Go through self-play. *Science*, **362**(6419), 1140–1144.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., *et al* (2017) Mastering the game of Go without human knowledge. *Nature*, **550**(7676), 354–359.

Szabó, B. and Babuška, I. (2021) *Finite Element Analysis: Method, Verification and Validation,* John Wiley & Sons, Hoboken, NJ.

Tang, B. (1993) Orthogonal array-based latin hypercubes. *Journal of the American Statistical Association*, **88**(424), 1392–1397.

Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M. and Solomon, J.M. (2019) Dynamic graph CNN for learning on point clouds. *ACM Transactions on Graphics*, **38**(5), 1–12.

Wang, Y. and Yue, X. (2024) Multimodal deep learning for manufacturing systems: Recent progress and future trends, in N. Gaw, P. M. Pardalos, M. R. Gahrooei (eds.), *Multimodal and Tensor Data Analytics for Industrial Systems Improvement*, Springer, Cham, Switzerland, pp. 221–252.

Wen, Y., Yue, X., Hunt, J.H. and Shi, J. (2018) Feasibility analysis of composite fuselage shape control via finite element analysis. *Journal of Manufacturing Systems*, **46**, 272–281.

Xing, Y. (2017) Fixture layout design of sheet metal parts based on global optimization algorithms. *Journal of Manufacturing Science and Engineering*, **139**(10), 101004.

Yang, B., Wang, Z., Yang, Y., Kang, Y. and Li, X. (2017) Optimum fixture locating layout for sheet metal part by integrating kriging with cuckoo search algorithm. *The International Journal of Advanced Manufacturing Technology*, **91**(1), 327–340.

Yue, X. and Shi, J. (2018) Surrogate model–based optimal feed-forward control for dimensional-variation reduction in composiste parts' assembly processes. *Journal of Quality Technology*, **50**(3), 279–289.

Yue, X., Wen, Y., Hunt, J.H. and Shi, J. (2018) Surrogate model-based control considering uncertainties for composite fuselage assembly. *Journal of Manufacturing Science and Engineering*, **140**(4), 041017.

Yue, X., Wen, Y., Hunt, J.H. and Shi, J. (2021) Active learning for Gaussian process considering uncertainties with application to shape control of composite fuselage. *IEEE Transactions on Automation Science and Engineering*, **18**(1), 36–46.

Zhang, T. and Shi, J. (2016) Stream of variation modeling and analysis for compliant composite part assembly—part ii: Multistation processes. *Journal of Manufacturing Science and Engineering*, **138**(12), 121004.

Zhong, Z., Mou, S., Hunt, J.H. and Shi, J. (2022) Finite element analysis model-based cautious automatic optimal shape control for fuselage assembly. *Journal of Manufacturing Science and Engineering*, **144**(8). 081009.

Zhong, Z., Mou, S., Hunt, J.H. and Shi, J. (2023) Convex relaxation for optimal fixture layout design. *IISE Transactions*, **55**(7), 746–754.