Proceedings of the Design Society, Volume 6: DESIGN 2026
https://doi.org/10.1017/pds.2026.10617

A deep reinforcement learning approach for the multi-objective, segment-based generative design of sheet metal components

Christoph Wittig Adão✉, Saruka Muralitharan, Jiahang Li, Markus Döllken and Sven Matthiesen

Karlsruhe Institute of Technology, Germany

✉ christoph.wittig@kit.edu

ABSTRACT: Current approaches for the generative design of sheet metal parts only take singular optimization goals into account. This paper presents a concept for a deep reinforcement learning approach to train an agent to generate sheet metal parts by combining segments from a predefined library. Through a weighted reward function, agents can be trained for different or combined optimization goals, such as weight, cost, or sustainability. The resulting agents enable the creation of a pareto front of optimal solutions, supporting efficient exploration of the design space for diverse design objectives.

KEYWORDS: design automation, computational design methods, design space exploration, artificial intelligence (AI), data-driven design

1. Introduction

As sustainability becomes an increasingly important design objective, the demand for developing and manufacturing environmentally responsible products continues to grow (Katsikeas et al., 2016). Due to their recyclability, efficient material usage, and structural performance, sheet metal parts play a key role in achieving sustainability goals during both manufacturing and product operation (Krause et al., 2025). The design and development of such sheet metal parts can be time-consuming and require extensive expertise, limiting the use of this manufacturing process (Doellken et al., 2021). Therefore, alternative design methods are being explored to accelerate and support the design process.

One method that has gained considerable attention in recent years is Generative Design. Regenwetter et al. define the generative design process as the creation of designs by algorithms with little or no human intervention (Regenwetter et al., 2022). Generative Design enables the automated exploration of large design spaces, accelerating the development process and reducing dependence on extensive expert knowledge (McKnight, 2017). Generative Design has already been successfully applied to a variety of manufacturing processes such as additive manufacturing or milling. However, its potential remains largely unexplored in the context of sheet metal design.

Only a few algorithms and research efforts address the specific challenges of generating manufacturable and mechanically feasible sheet metal parts (Barda et al., 2023; Patel & Campbell, 2010). Although these approaches already generate a large solution space, the problem is that only individual optimization goals are pursued when searching this solution space, and the search algorithms are not capable of learning. This paper contributes to narrowing down the large solution space of sheet metal segments with an approach for training a reinforcement learning agent through feedback and transferring this knowledge to new design cases. Unlike previous search-based methods that focus on single optimization goals, the proposed approach enables scalable and adaptive exploration of the design space across multiple, potentially conflicting objectives, such as weight, cost, and sustainability.

https://doi.org/10.1017/pds.2026.10617 Published online by Cambridge University Press

2. Related work

2.1. Generative design in engineering design

To establish the foundation for the proposed approach, this section reviews existing research on generative design, which has become an important research field in engineering design in recent years. A large body of work in generative design focuses on automating and accelerating the design process using algorithms that generate, evaluate, and refine design solutions (Oh et al., 2019). These approaches can be classified according to their methodological orientation, objectives, and application domains (Mountstephens & Teo, 2020).

Common methods include computer aided design (CAD) based design generation, such as generating 3D models from sketches, from sequences of CAD commands, from point clouds, or even from images (Heidari & Iosifidis, 2025). These approaches primarily focus on the geometric design process itself and often disregard the specific requirements and constraints of manufacturing processes. When manufacturing is considered, the focus is mostly on additive manufacturing or common conventional processes such as CNC milling.

For example, the work of Ertelt and Shea explores generative design for milling processes using a shape grammar approach. The approach is simplified: it assumes a single surface, constant cutting depth, and discrete movement directions. The main focus lies in transferring the generated design to feasible manufacturing paths, rather than exploring the design space itself. (Ertelt & Shea, 2008)

In contrast, other approaches focus on feature-based CAD modification (Krish, 2011). While such methods can generate valid designs efficiently, they often optimize for a single objective, such as mass reduction, and therefore do not sufficiently explore the broader design space - a key characteristic of true generative design.

First approaches to generative design for sheet metal components have been proposed by Patel and Campbell as well as by Barda et al. Patel and Campbell developed a shape grammar approach where operators iteratively modify a flat 2D surface to generate a 3D sheet metal component. Unsuitable solutions are pruned through rule-based filtering, limiting exploration but ensuring manufacturability. (Patel & Campbell, 2010)

Barda et al. proposed a two-step process: in the first step, individual sheet metal segments are generated and stored in a segment library. In the second step, combinations of these segments are searched to form complete components. However, the combinatorial space grows exponentially with the number of possible segment combinations. Exhaustively simulating all possible solutions is computationally infeasible. Thus, Barda et al. applied a beam search method to pre-select promising candidates for further evaluation. (Barda et al., 2023)

Beam search, however, is a deterministic search strategy and has several drawbacks in this context. It considers only the top-k most promising candidates in each step and therefore may discard potentially optimal long-term solutions early in the search. Furthermore, it does not learn from past iterations and quickly becomes computationally expensive in large state spaces despite pruning.

2.2. Reinforcement learning algorithms in design tasks

To overcome these limitations, reinforcement learning offers a promising alternative. reinforcement learning can learn from interaction with the environment and balance exploration and exploitation, allowing them to discover unconventional but potentially optimal design paths (Kaelbling et al., 1996). Moreover, reinforcement learning agents can handle large and complex design spaces, for example by generalizing through neural network representations. An agent in this context is the decision-making entity within the reinforcement learning framework that interacts with the environment by selecting actions, receiving feedback, and improving its policy over time. In each step, the agent observes the current state, performs an action based on a policy, and receives a reward that reflects the quality of the action in that state. The goal of reinforcement learning is to find an optimal policy that maximizes the expected cumulative reward (Tan, 2021).

Deep reinforcement learning extends this concept by integrating deep neural networks to approximate the policy or value function. This enables the algorithm to handle high-dimensional, complex, and sequential decision problems in which the optimal action depends on previous states (Plaat, 2022).

In engineering design, reinforcement learning and deep reinforcement learning have already been successfully applied to various tasks. Applications include fluid dynamics control (Rabault et al., 2019),

https://doi.org/10.1017/pds.2026.10617 Published online by Cambridge University Press

topology optimization (Sun & Ma, 2020) or vehicle energy management (Liessner et al., 2019). In these domains, reinforcement learning has shown strong performance in optimizing multiple conflicting objectives and discovering control or design strategies beyond human intuition.
The selection of a suitable reinforcement learning strategy depends on criteria such as the type of action space, data availability, training stability, and sensitivity to hyperparameters. Several strategies exist, which can be categorized according to their structure and application context (Plaat, 2022):

- Action space: Algorithms can operate in discrete or continuous action spaces, which may be one- or multi-dimensional.
- Model-based vs. model-free: Model-based approaches rely on a known model of the environment, while model-free algorithms, such as Proximal Policy Optimization (PPO) or Advantage Actor–Critic (A2C), learn directly from experience without requiring such a model.
- On-policy vs. off-policy: In on-policy algorithms (e.g., PPO), the agent learns from actions generated by its current policy; off-policy methods (e.g., DDPG) learn from a replay buffer containing past experiences.
- Exploration strategy: Different algorithms apply distinct exploration mechanisms, such as random noise (DDPG) or probabilistic action selection (DQN).

Deep reinforcement learning has also proven effective for combinatorial optimization problems, where candidate solutions are constructed from discrete building blocks. Examples include the Traveling Salesman Problem (TSP) and multi-objective combinatorial optimization (Wang et al., 2024). Reinforcement learning-based methods can learn heuristics that construct or improve solutions iteratively, often outperforming traditional hand-crafted algorithms, especially for large or complex search spaces.
Analogously, sheet metal component generation can be formulated as a combinatorial optimization in the design space of possible segment configurations. The approach of Barda et al. (2023) - generating and combining individual sheet metal segments - fits this formulation well and motivates the use of reinforcement learning to explore the combinatorial design space efficiently while considering multiple objectives such as weight, manufacturability, and sustainability. Despite its potential, no reinforcement learning-based approach for combining sheet metal segments has been presented in the current state of research.

2.3. Research question

The literature highlights a gap in the systematic exploration of the design space for generative design of sheet metal components. Existing deterministic search methods lack the ability to learn and typically optimize only a single objective. Reinforcement Learning provides a promising solution to these limitations. Reinforcement learning agents learn through interaction with their environment and balance exploration with exploitation, enabling the discovery of unconventional yet potentially optimal design solutions. By leveraging neural network representations, reinforcement learning agents can manage large and complex design spaces, making them particularly well-suited for addressing combinatorial design challenges such as the combination of sheet metal segments in generative design.
The aim of this paper is to develop an approach for training a reinforcement learning agent that enables efficient exploration of the combinatorial design space of sheet metal components while simultaneously considering multiple, potentially conflicting, optimization objectives. This aim leads to the following research question:
How can a reinforcement learning agent be designed to learn the selection and combination of sheet metal segments while considering multiple optimization objectives?

3. Developed approach

Building on the identified research gap, this chapter introduces the proposed approach for training a Reinforcement Learning agent that autonomously learns to combine sheet metal segments from a library to generate feasible designs while efficiently exploring the design space across multiple, potentially conflicting optimization objectives. In principle, various methods, such as rule-based, machine-learning-based, or deep-learning-based techniques, can be employed to generate segments to fill this library. By combining the segments from the library, the strengths of individual geometric or functional features can be integrated, potentially leading to better-performing sheet metal components.

https://doi.org/10.1017/pds.2026.10617 Published online by Cambridge University Press

Building upon this idea, we developed an approach that aims to select suitable segments from a predefined library to form a complete sheet metal part, taking into account structural requirements. The resulting parts are designed to be manufacturable from a single sheet using standard bending operations. It should be emphasized that the proposed algorithm has not yet been implemented or validated experimentally; thus, this work presents a conceptual framework intended as a foundation for future development.

To address the combinatorial nature of the problem, our method employs deep reinforcement learning. The agent interacts with an environment and receives feedback via a reward function, which guides it toward optimal combinations of sheet metal segments. After training, the agent is expected to generalize across different design cases. Furthermore, by varying the weights in the reward function, multiple agents can be trained to pursue different optimization objectives - such as weight reduction, stiffness improvement, or manufacturing simplicity.

In a preceding step, a suitable segment library is generated, containing all feasible segments that connect given mounting points (Figure 1). The segment library can be generated using the approaches presented in the related work section, thereby leveraging and combining the complementary strengths of these methods. These segments represent partial solutions and can be combined in numerous ways, resulting in a large design space. Since all segments share predefined connection interfaces, the segment combination can be performed via Boolean operations on the segment geometries.

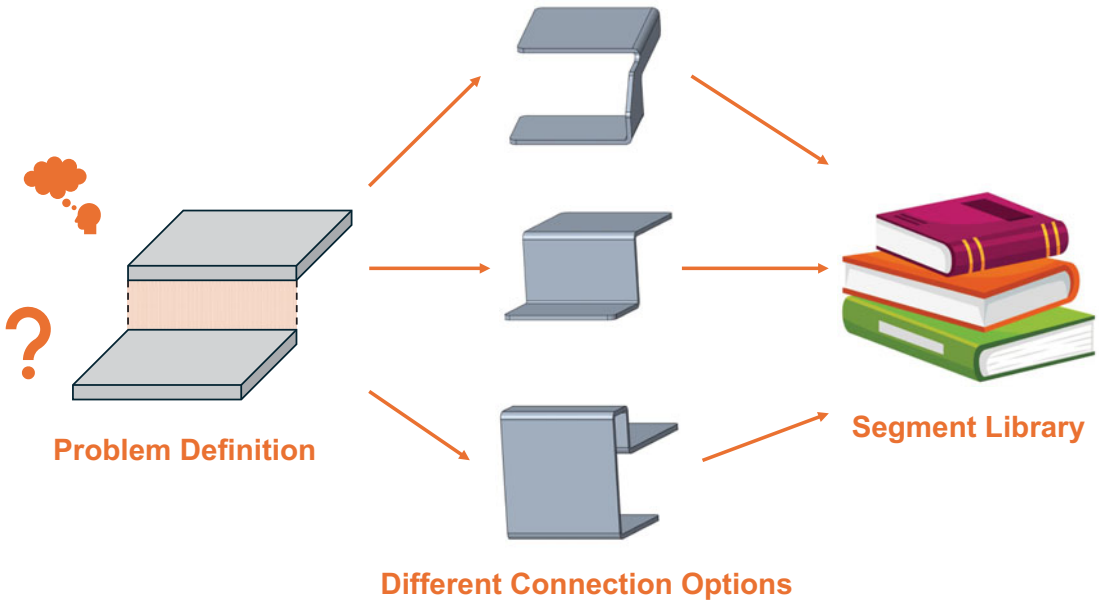


Figure 1. Schematic representation of a segmented sheet metal design problem with solution segments and the library for storing segments

To build a functional deep reinforcement learning framework, both the agent and the environment must be defined. In our approach, the agent sequentially selects sheet metal segments from the library. The environment then combines these segments into a single continuous sheet metal structure and provides feedback based on the resulting design's performance and manufacturability.

The design generation problem of combining different segments to a complete part can be formulated as a graph-based combinatorial problem with a variable graph size, as pictured in Figure 2. Therefore, we utilize a Graph Neural Network (GNN) as part of the policy model. A key advantage of GNNs is their ability to handle graphs of arbitrary size and topology - meaning that the input does not need to have a fixed dimension. This makes them particularly well-suited for representing structural assemblies, where the number and connectivity of segments can vary between designs.

To specify this sequential decision-making process, the four main components of a reinforcement learning setup - **state, action, reward, and training** - must be defined. These elements are described in detail in the following sections.

https://doi.org/10.1017/pds.2026.10617 Published online by Cambridge University Press

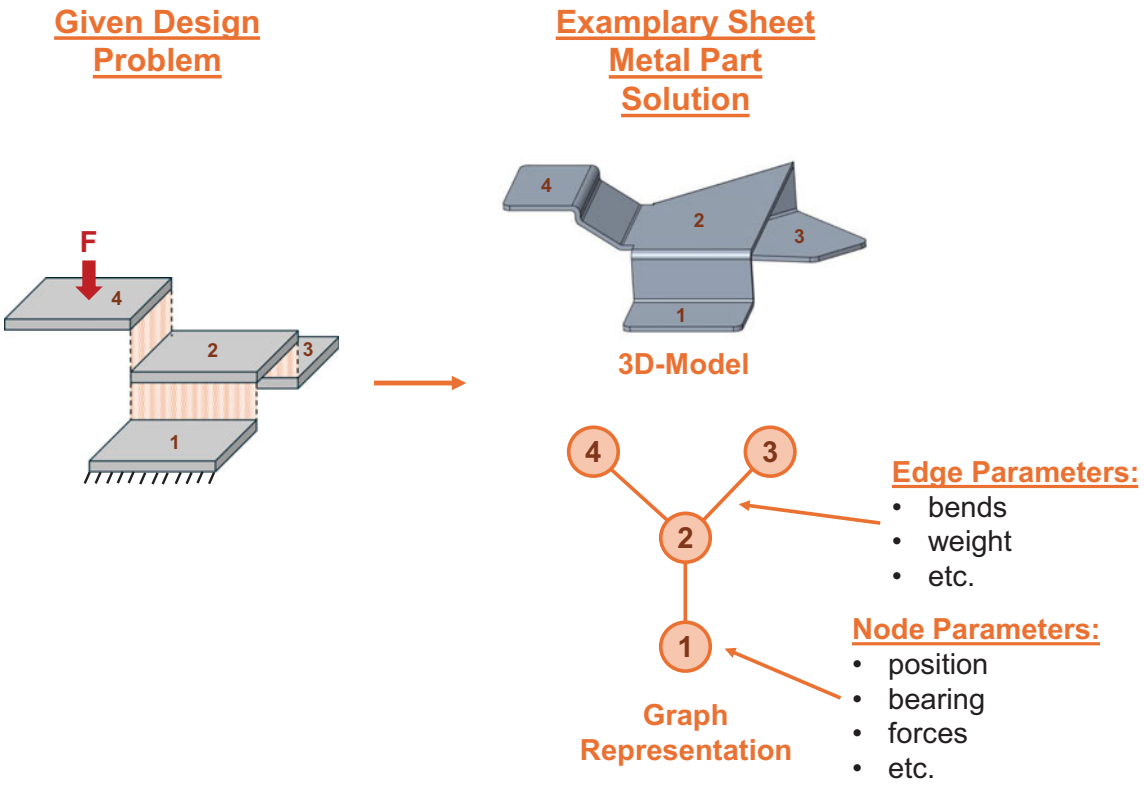


Figure 2. Transferring a design problem from the sheet metal area to a graph that can be processed by the GNN, with edge and node parameters representing the sheet metal part

3.1. State

The state is represented by a graph that evolves dynamically throughout the design process. In this graph, the nodes correspond to the mounting points, while the edges represent the sheet metal segments connecting them. An example of a sheet metal component and its corresponding graph is shown in Figure 2.

Each node contains the spatial coordinates of the corresponding mounting point as well as a binary indicator specifying whether this point has already been connected. Each edge stores information about the two connected nodes, the geometrical properties of the segment (such as length, cross-sectional area, and thickness), and the material properties (such as weight and carbon footprint). Additionally, each edge includes a flag indicating whether the segment has already been built into the component.

This representation enables the model to capture the current design state of the component at any given time, including both geometric relationships and material-related attributes.

3.2. Action

As described earlier, the agent is responsible for selecting actions during the construction process. Each action corresponds to either the selection of a specific segment from the library or a dedicated "stop" action. When the agent determines that the component is complete, it selects the "stop" action, signaling the end of the construction sequence.

To ensure manufacturability, the agent is not allowed to select actions that would lead to collisions between already built-in segments. Therefore, a collision check is performed after each step. If a collision is detected, the termination condition is triggered, and the current episode ends prematurely.

An episode also ends when the agent chooses the "stop" action, after which a new episode can begin. Through this process, the agent iteratively learns which segment combinations lead to valid and optimal designs.

3.3. Reward

The reward function provides feedback to the agent after each action and is designed to guide it toward optimal design configurations.

https://doi.org/10.1017/pds.2026.10617 Published online by Cambridge University Press

During the early design phase, when not all mounting points are connected, small positive rewards are given for newly connected points. The magnitude of the reward increases as the component approaches the defined optimization objectives. As example objectives, designs that are lighter, require fewer bends, or use less material receive higher rewards; moreover, weighted combinations of these objectives can be considered to reflect multi-objective design preferences. This helps the agent learn which design directions are beneficial.

At the end of each episode, a final reward is computed based on key performance indicators such as the results of finite element analysis, manufacturing costs, and the carbon footprint of the completed component.

If an episode ends due to the termination condition being met (e.g., a collision or invalid geometry), the agent receives a negative reward. This penalization ensures that the agent learns to avoid such poor design decisions in future episodes.

3.4. Training

The training of the agent is realized using an actor–critic algorithm, which consists of two neural networks: the actor network and the critic network. In our approach, both are implemented as a GNN. An overview of the training process is shown in Figure 3.

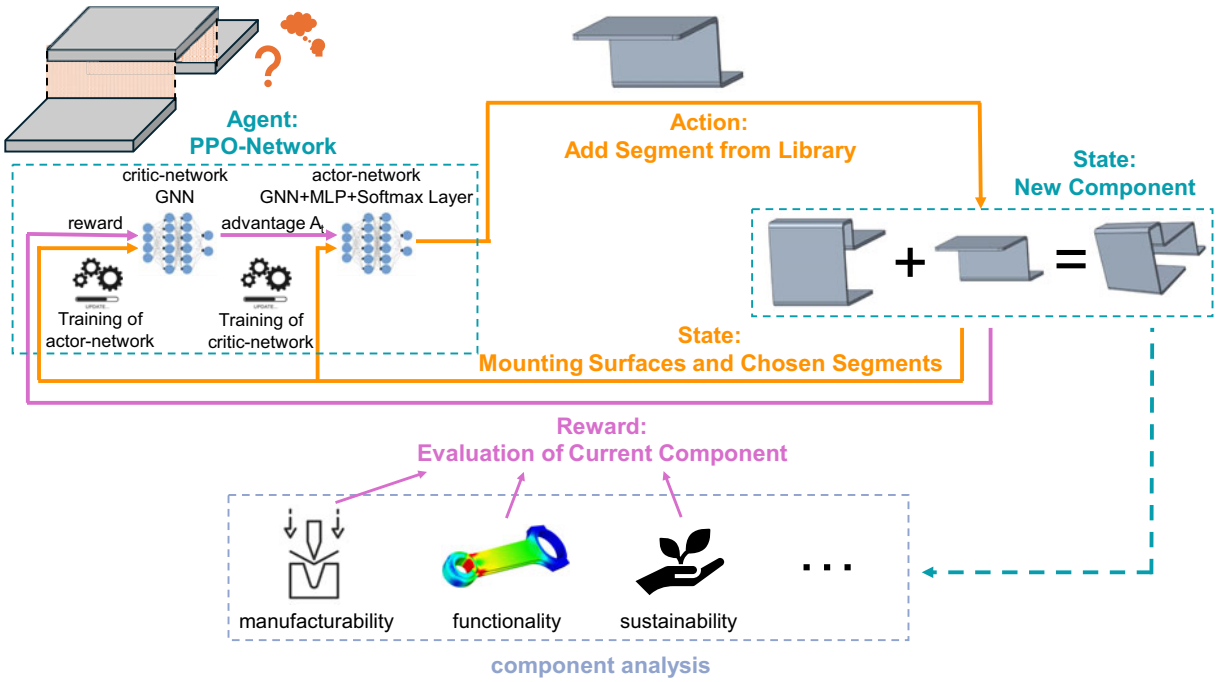


Figure 3. Schematic representation of the training of the proposed reinforcement learning-based design approach. The agent selects a new segment from the library as its action, which updates the current component and generates a new state. A reward function evaluates this new state based on multiple optimization objectives and, together with the current state, provides feedback to the agent, enabling it to learn and improve its segment selection strategy over time

At the beginning of each episode, the initial state of the environment is passed to the algorithm. The actor network represents the policy, which selects actions for the current state. First, the GNN processes the entire design graph - consisting of all mounting points and available segments—to compute node and edge embeddings. These embeddings encode the structural context, capturing relationships between nodes and edges within the graph. After training, the GNN can generate context-sensitive embeddings that reflect both local and global structural dependencies.

Subsequently, the edge embeddings are passed to a multilayer perceptron, which outputs action values that are converted into probabilities using a Softmax layer. These probabilities define the likelihood of selecting each segment or the "stop" action. Actions are selected iteratively: if there are ten nodes, the

https://doi.org/10.1017/pds.2026.10617 Published online by Cambridge University Press

same operation is applied sequentially ten times, enabling the model to handle graphs of variable size. This makes the approach scalable and invariant to the number of nodes and edges.

When the agent selects a new segment, the environment updates the current component by adding the corresponding geometry. The reward for this action is then computed and passed, together with the new state, to the actor–critic algorithm. The critic network estimates the expected return of the current graph state and updates its weights by minimizing the error between the predicted value and the observed reward. From this, the critic calculates the advantage, which is used to update the actor's parameters to improve its policy.

This process is repeated until the agent chooses the "stop" action or the termination condition is reached. Over many episodes, the actor network learns to select actions that maximize the expected reward, while the critic network becomes better at estimating future returns.

To ensure an effective training process, a balance between exploration and exploitation is maintained. The Softmax output provides a probability distribution over all possible actions. During exploitation, the action with the highest probability is chosen. During exploration, random actions are selected with a certain probability, allowing the agent to discover new and potentially better strategies. In the early training phase, exploration is emphasized; as training progresses, exploitation becomes dominant. This balance enables the agent to acquire comprehensive knowledge about the design space and later refine its decisions toward optimal solutions.

To promote generalization, each episode features varying load cases, different arrangements and numbers of mounting points, and diverse segment geometries. Consequently, the agent is not specialized for a single configuration but can handle a broad range of use cases.

Given the graph-based and continuous nature of the problem and the absence of a known environment model, suitable deep reinforcement learning algorithms include PPO and A2C. As PPO is widely used for combinatorial problems and exhibits stable training behavior across hyperparameter settings, it was selected for this work.

4. Application for design space exploration in generative design

By adjusting the weights in the reward function, agents can be trained not only for individual optimization goals but also for combined or weighted trade-offs between multiple objectives - for example, balancing weight reduction, manufacturing cost, and sustainability. Each agent thus explores a specific region of the multi-objective design space, reflecting a different compromise between competing criteria.

The collection of trained agents generates a diverse set of feasible design candidates, which together form a Pareto front representing the spectrum of optimal trade-offs between the defined objectives. This Pareto front provides an overview of the achievable solution space and enables designers to understand how improving one criterion may affect others, such as the relationship between stiffness and material usage. In this way, the proposed approach supports systematic and automated design space exploration. Instead of relying on manual parameter adjustments or exhaustive searches, the reinforcement learning agents learn to focus their exploration on promising and manufacturable design regions based on the feedback provided by the reward function.

This process is illustrated in Figure 4, which summarizes how the combination of multiple trained agents leads to a structured exploration of the design space and supports the identification of optimal trade-offs between design objectives.

5. Discussion

To answer the research question "How must a Reinforcement Learning agent be designed to select and combine sheet metal segments while considering multiple optimization objectives?" the main aspects of the agent, namely state, action, reward and training, are defined and described in detail. The presented approach offers the possibility of developing an agent that selects suitable combinations of sheet metal segments for various optimization goals. Compared to the approach of Barda et al. (2023), different optimization goals can be considered and pursued simultaneously. By using multiple agents that are trained for distinct or combined reward functions, optimal results for diverse use cases can be generated. Consequently, the generalizability and adaptability of the generated design solutions are increased.

https://doi.org/10.1017/pds.2026.10617 Published online by Cambridge University Press

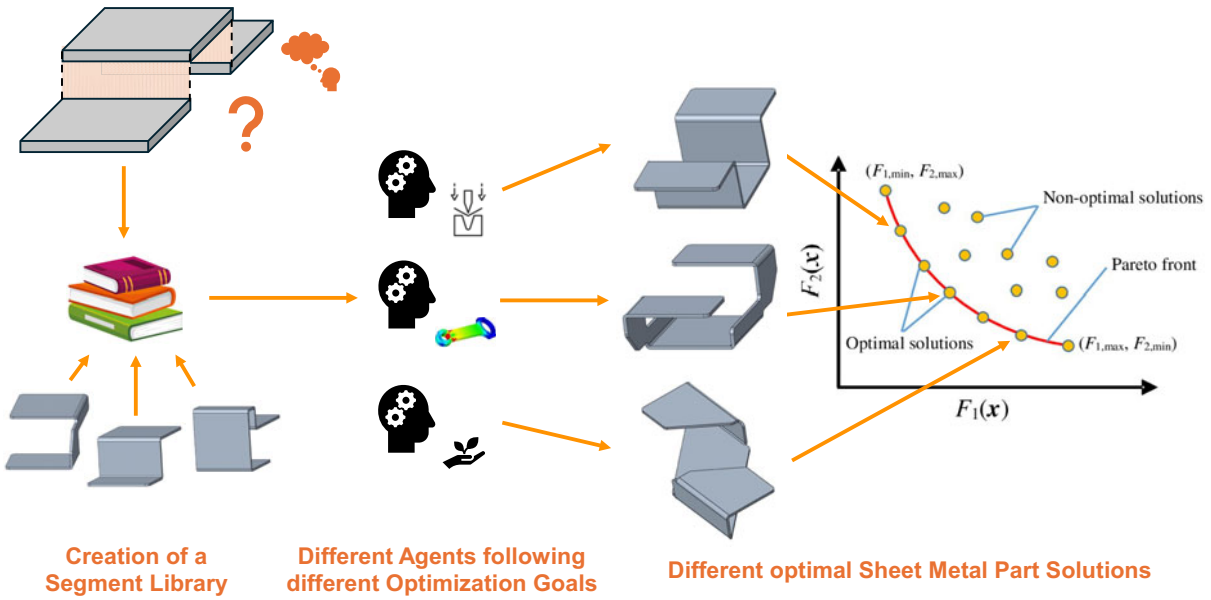


Figure 4. With several reinforcement learning agents trained on different weightings of the optimization goals, a Pareto front of optimal design concepts can be generated

Beyond computational efficiency, the agent learns design strategies autonomously through interaction with the environment. This learning-based exploration of the design space enables the discovery of unconventional, yet feasible, sheet metal configurations that may be overlooked by traditional optimization or rule-based design approaches. The integration of design knowledge into a reinforcement learning framework bridges the gap between algorithmic exploration and human-driven engineering intuition.

As deep reinforcement learning agents learn policies based on the received rewards, the generation of solutions is often more robust against poor initial heuristics. Further advantages include the adaptability to dynamic or uncertain environments and the suitability for complex, graph-based design representations.

As mentioned in Chapter 3, the main limitation of this work is that no implementation of the presented approach has yet been realized. This currently limits the significance of the results. Implementing the algorithm is expected to yield new insights into the state, action, reward, and other individual components of the approach. These findings may lead to modifications or refinements of the proposed structure in future work.

Additionally, a sufficient characterization of the sheet metal segments within the library can pose a challenge. Segments with different geometries may, for instance, share identical weights and numbers of bends. Hence, it is crucial to describe the segments comprehensively with suitable features, as their geometric variations can strongly influence their mechanical properties. The impact of the segment characterization on the learned policy and resulting design quality must be systematically investigated to determine which geometric, mechanical, and manufacturing-related descriptors are most suitable for representing the segments.

6. Conclusion and outlook

This paper presented an approach for using Deep Reinforcement Learning to improve the generative design of sheet metal components. The approach introduces a reinforcement learning agent capable of selecting and combining suitable segments from a predefined segment library to form manufacturable sheet metal parts. By defining the design problem as a graph-based combinatorial task and utilizing a GNN for state representation, the method enables the handling of variable-sized design configurations and complex interdependencies between components.

Through varying the weights in the reward function, different agents can be trained to optimize for individual or combined design objectives. This allows the generation of a Pareto front of solutions that balance competing goals such as weight, stiffness, and sustainability. The developer can then select the most suitable design based on specific application requirements. In doing so, the approach provides a

https://doi.org/10.1017/pds.2026.10617 Published online by Cambridge University Press

structured way to explore the design space efficiently while maintaining flexibility for different optimization goals.

The immediate next step is the implementation and experimental validation of the proposed approach on representative use cases. Implementing the algorithm will provide insights into the definition of states, actions, and reward functions and may lead to refinements of the proposed structure. Furthermore, a comprehensive characterization of the segment library will be required to ensure that geometrical and mechanical variations are accurately represented.

In the medium term, the method can be extended to additional manufacturing processes such as welding or deep drawing, thereby expanding the applicability beyond pure bending operations. Moreover, the transferability of the underlying principles to other modular or rule-based design domains, such as lightweight structures or modular assemblies, can be investigated to assess the generality of the approach.

In the long term, the approach could be expanded toward cooperative multi-agent systems, where multiple agents pursue different or weighted optimization objectives and jointly explore trade-offs in the design space. Such systems could enable the automated generation of Pareto-optimal design variants considering stiffness, weight, cost, and environmental impact. Integrating knowledge-based engineering methods or constraint solvers within the learning framework represents another promising research direction to improve robustness and industrial applicability.

References

Barda, A., Tevet, G., Schulz, A., & Bermano, A. H. (2023). Generative Design of Sheet Metal Structures. *ACM Trans. Graph.*, 42(4), 116:1–116:13. https://doi.org/10.1145/3592444

Doellken, M., Arndt, L., Nelius, T., & Matthiesen, S. (2021). Identifying an opportunistic method in design for manufacturing: An experimental study on successful a on the manufacturability and manufacturing effort of design concepts. *Procedia CIRP*, 100, 720–725. https://doi.org/10.1016/j.procir.2021.05.090

Ertelt, C., & Shea, K. (2008). Generative Design and CNC Fabrication Using Shape Grammars. *Volume 1: 34th Design Automation Conference, Parts A and B*, 25–34. https://doi.org/10.1115/DETC2008-49856

Heidari, N., & Iosifidis, A. (2025). Geometric Deep Learning for Computer-Aided Design: A Survey. *IEEE Access*, 13, 119305–119334. https://doi.org/10.1109/ACCESS.2025.3587121

Kaelbling, L. P., Littman, M. L., Moore, A. W., & Hall, S. (1996). Reinforcement Learning: A Survey. *Reinforcement Learning*. https://doi.org/10.1613/jair.301

Katsikeas, C. S., Leonidou, C. N., & Zeriti, A. (2016). Eco-friendly product development strategy: Antecedents, outcomes, and contingent effects. *Journal of the Academy of Marketing Science*, 44(6), 660–684. https://doi.org/10.1007/s11747-015-0470-5

Krause, A., Dielhenn, J., Jess, S., Wagenmann, S., Doellken, M., Albers, A., Matthiesen, S., & Bursac, N. (2025). Sustainable Sheet-Metal Design: Employing the Product Carbon Footprint as Support for Engineers in Developing New Product Generations. *Procedia CIRP*, 136, 510–515. https://doi.org/10.1016/j.procir.2025.08.088

Krish, S. (2011). A practical generative design method. *Computer-Aided Design*, 43(1), 88–100. https://doi.org/10.1016/j.cad.2010.09.009

Liessner, R., Schmitt, J., Dietermann, A., & Bäker, B. (2019). Hyperparameter Optimization for Deep Reinforcement Learning in Vehicle Energy Management: Proceedings of the 11th International Conference on Agents and Artificial Intelligence, 134–144. https://doi.org/10.5220/0007364701340144

McKnight, M. (2017). Generative Design: What it is? How is it being used? *Why it's a game changer. KnE Engineering*, 176–181. https://doi.org/10.18502/keg.v2i2.612

Mountstephens, J., & Teo, J. (2020). Progress and Challenges in Generative Product Design: A Review of Systems. *Computers*, 9(4), Article 4. https://doi.org/10.3390/computers9040080

Oh, S., Jung, Y., Kim, S., Lee, I., & Kang, N. (2019). Deep Generative Design: Integration of Topology Optimization and Generative Models. *Journal of Mechanical Design*, 141(11), 111405. https://doi.org/10.1115/1.4044229

Patel, J., & Campbell, M. I. (2010). An Approach to Automate and Optimize Concept Generation of Sheet Metal Parts by Topological and Parametric Decoupling. *Journal of Mechanical Design*, 132(051001). https://doi.org/10.1115/1.4001409

Plaat, A. (2022). *Deep Reinforcement Learning*. Springer Nature Singapore. https://doi.org/10.1007/978-981-19-0638-1

Rabault, J., Kuchta, M., Jensen, A., Réglade, U., & Cerardi, N. (2019). Artificial neural networks trained through deep reinforcement learning discover control strategies for active flow control. *Journal of Fluid Mechanics*, 865, 281–302. https://doi.org/10.1017/jfm.2019.62

https://doi.org/10.1017/pds.2026.10617 Published online by Cambridge University Press

Regenwetter, L., Nobari, A. H., & Ahmed, F. (2022). Deep Generative Models in Engineering Design: A Review. *Journal of Mechanical Design*, 144(071704). https://doi.org/10.1115/1.4053859

Sun, H., & Ma, L. (2020). Generative Design by Using Exploration Approaches of Reinforcement Learning in Density-Based Structural Topology Optimization. *Designs*, 4(2), 10. https://doi.org/10.3390/designs4020010

Tan, H. (2021). Reinforcement Learning with Deep Deterministic Policy Gradient. *2021 International Conference on Artificial Intelligence, Big Data and Algorithms (CAIBDA)*, 82–85. https://doi.org/10.1109/CAIBDA53561.2021.00025

Wang, Z., Yao, S., Li, G., & Zhang, Q. (2024). Multiobjective Combinatorial Optimization Using a Single Deep Reinforcement Learning Model. *IEEE Transactions on Cybernetics*, 54(3), 1984–1996. https://doi.org/10.1109/TCYB.2023.3312476

https://doi.org/10.1017/pds.2026.10617 Published online by Cambridge University Press