Solid Freeform Fabrication 2023: Proceedings of the 34th Annual International Solid Freeform Fabrication Symposium – An Additive Manufacturing Conference
Reviewed Paper

Methodical Approach to Reducing Design Time by using Neural Networks in Early Stages of Concept Development

M.Ott*, N. Meihöfener*, I. Mozgova*

Data Management in Mechanical Engineering (DMB), Direct Manufacturing Research Center (DMRC), Faculty of Mechanical Engineering, Paderborn University, 33098 Paderborn, Germany

Abstract

Modern companies often face various challenges in concept development of products or systems. Design engineers prepare initial concepts as 3D models. These are then simulated by computational engineers. If requirements are not met, this necessitates an iterative process that runs between the design and computation departments until a valid concept is created. Design methods such as topology optimization are often used here. The upcoming result is then attempted to be adapted to certain manufacturing processes. These iteration loops can sometimes take a very long time, since the model construction and structural optimization generate large computational efforts. The present work shows on an example a methodical approach, which represents a first proof of concept, to solving this problem, including a description of methods and techniques, as well as possible problems in a detailed analysis concerning training data for neural networks and their abstraction capabilities. It is evident that additional research work needs to be conducted for further utilization in order to address all arising questions.

Introduction to product development processes and design stages

Design tasks in current development processes have evolved historically and are highly diverse. Therefore, new concepts and methods need to be developed, especially in the fields of design methodology, topology optimization, and machine learning. While manufacturing processes and computer performance were limited in the past, today we witness extraordinary manufacturing possibilities and the ability to simulate them virtually. However, achieving this requires the establishment of new workflows and information flows. Consequently, it is crucial to first understand the fundamental principles of current process models in order to ensure a solid foundation.

The most renowned design process model originates from Pahl and Beitz (see Figure 1), This model divides the design process into four distinct phases [1], which largely align with the contemporary VDI2221-1(2019) standard in principle:

- Planning,
- Concept,
- Draft,
- Finalize.

Each of these phases entails different information, which result in various tasks (see Figure 1). Along with these differences, additional or limited opportunities arise for employing different techniques. For instance, in early phases such as planning and concept development, written documentation including requirement lists, product proposals, or sketches of principle solutions may still be prevalent. Conversely, in the detailed design phases, it is inconceivable to work without Computer-Aided Design (CAD) systems. As a result, modern organizations typically have a structure that distributes these tasks among various departments. Some departments contribute

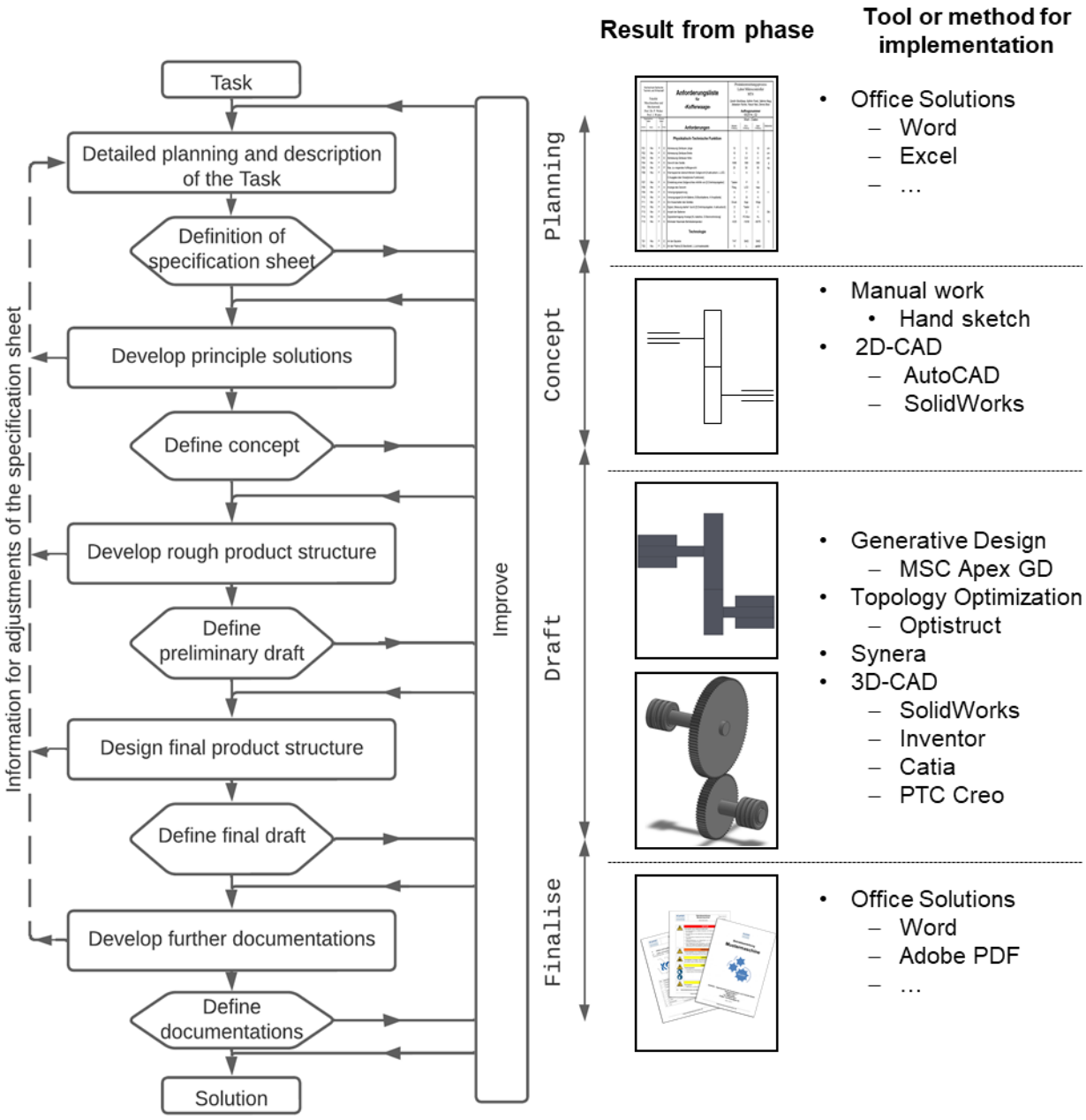

Figure 1: Procedure model in construction according to [1]

design drafts and proposals, while a dedicated analysis department employs diverse Finite Element Methods (FEM) to examine these designs for stress concentrations. This often leads to complex iteration loops that involve problem identification and solution generation, with different departments handling the problem description and its corresponding solution [2,3]. The interplay between problem description and the proposed solution [4] is illustrated in Figure 2. These problem solutions can be initially categorized into three groups, as outlined in [5]:

- Progressive iterations,
- Correcting iterations,
- Coordinating iterations.

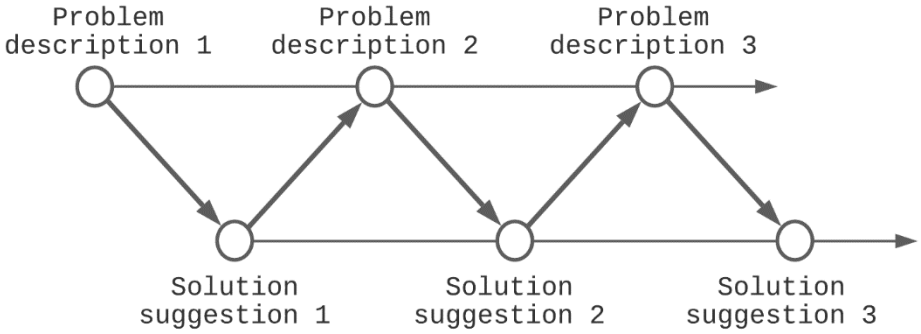

Figure 2: Coevolution of problem and solution according to [4]

This classification is crucial for minimizing unnecessary iterations while allowing necessary, meaningful, and improvement-focused iterations. The basic manifestations of iterations in product development, as described in [5] can initially be categorized into the aforementioned overarching groups. Within the context of this study, the objective of the proposed methodology is to reduce development time by utilizing neural networks in the early stages of product development. However, due to the prevalent existence of concepts or rough designs during these early phases, which progressively become more concrete, refined, or converge towards an optimal solution throughout the process, it is recommended to focus on one of the manifestations mentioned within the methodology.

Since the developed method can be seen in early phases, but quite as an intersection between different ones, here the cyclic strategy is seen as helpful. This cyclic view has already found its application in historical context which was developed for example after Wögerbauer [6] in the year 1942, since even there the iterative problem solutions were seen as meaningful means for the improvement.

Problems and need of action

Topology optimization is another way of optimizing structures, alongside shape and parameter optimization. There are several algorithms available for this purpose, each with its own advantages and disadvantages. One such algorithm is the Solid Isotropic Material with Penalization (SIMP) algorithm, which subdivides a geometry into small elements, similar to a finite element method. The algorithm assigns a normalized modulus of elasticity to these elements. The density of the material, ranging from 0 to 1, is determined based on the ratio of the assigned modulus to the material properties. The algorithm penalizes low densities exponentially, hence its name. Since the algorithm does not delete elements, all elements need to be considered at each iteration. Performing topology optimization can be computationally expensive, especially for precise calculations and larger models, and it is often used primarily for concept generation. [7]

Many software tools used in the current calculation process require specialized knowledge, which is why topology optimization is typically performed in separate departments. As mentioned at the beginning of this work, iteration loops between design and analysis occur frequently and consume a significant amount of time.

In addition to iteration loops, the use of topology optimization results in a pre-final and not ready to build construction. This means that when directly transferred to a CAD system, the resulting geometry lacks parametric construction as a design basis, making changes without recomputation

challenging. Particularly when there are changes in requirements, such as manufacturing processes, this incurs additional effort [8].
Various methods for reverse engineering pose an ongoing research topic, although current methods already yield promising results, occasionally still prone to errors. In addition to automated methods, semi-automated or manual approaches are possible. Semi-automated methods, employing feature detection, can introduce initial parametric elements but do not operate entirely parametrically. This further highlights that topology optimization is primarily used for concept development. [9]

Apart from topology optimization, different design proposals can also be generated through generative design. However, these designs are not necessarily based on finite element analysis (FEA), which does not guarantee mechanical accuracy as a prerequisite. However, by eliminating these calculations, a significant time saving in the generation process can be achieved. The combination of the advantages of topology optimization and generative design, possibly using neural networks, has already been investigated in [10]. However, when using a neural network, there is a strong dependence on training data, and only a certain level of abstraction is possible. To effectively address these challenges, the utilization of designer-generated data, calculations, and neural networks offers various possibilities. Computer scientists, designers, and computational engineers may pursue different approaches to finding solutions, which is why there is a convergence of the three disciplines: computer science, design, and computation/simulation. However, when using a neural network, there is a strong dependence on training data, and only a certain level of abstraction is possible. To effectively address these challenges, the utilization of designer-generated data, calculations, and neural networks offers various possibilities. Computer scientists, designers, and computational engineers may pursue different approaches to finding solutions, which is why there is a convergence of the three disciplines: computer science, design, and computation/simulation.
Based on the explanations provided regarding the product development process and problem description, the main needs for action can be formulated as follows:

- Accurate and well-founded integration of the developed method into the product development process,
- Consideration of mechanical constraints and specifications,
- Rapid adaptation capabilities without additional effort,
- Generic applicability (independent of specific components),
- Reduction of time compared to conventional methods,
- Minimal user expertise required for broad application.

Methodical approach

The overall objective of this work is to achieve a design time reduction by developing a methodical approach combining the advantages of topology optimization, which mainly provides mechanically correct results, and those of generative design, in the form of fast geometry suggestions, by using neural networks in early stages of the concept development. As the product development process encompasses diverse information, data, and processes in its different phases (see Figure 1), it is essential to initially define the limitations and time constraints within the product development process, which then serve as the basis for the methodological approach.

The chronological sequence of the process, according to VDI 2221, is depicted using Business and Process Modeling and Notation (BPMN) 2.0 in Figure 4, where the gray-shaded "Design" field marks the timing and deployment of the method. BPMN 2.0 provides various elements for constructing processes, including events (start and end of process steps), gateways (X, OR, and XOR decisions), activities (process steps), and documents. Particularly useful is the division into individual lanes, which can represent different departments within a company. Therefore, the different phases of the process are divided into these lanes. [11] This includes the phases planning, concept, draft and finalize.

The method to be developed is positioned as an interface between the concept and draft phase. At this stage, there are already initial functional elements in the form of conceptual solutions or preliminary sketches. Simultaneously, there are requirements in the form of requirement lists or categorizations known through software tools such as DOORS or similar systems. This means that initially only rough geometries that represent a possible design space are available. However, these can be optimally utilized for topology optimizations. Rudimentary geometries initially require minimal effort, yet it is necessary to provide the optimization process with the maximum design space in order to achieve an optimal result. If the developed component is already highly detailed at this stage, the optimization step would not yield significant benefits as the possibilities for adapting the topology would be severely limited. Figure 3 illustrates a possible result of optimization. [12]

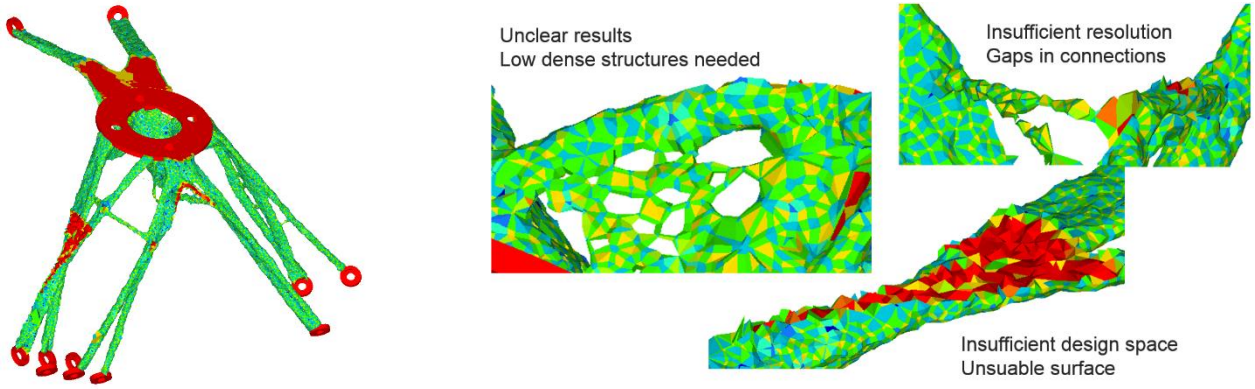

Figure 3: Possible results and limitations of topology optimization processes according to [13]

It can be observed that typical outcomes may contain various errors, which often require reconstruction. This reconstructed geometry is then provided to a structural engineer for verification using FEM analysis. The results are compared against norms and requirements, and any deviations are communicated back to the designer. The designer makes the necessary changes and generates new designs, which are then subjected to this loop again. These iterations can be time-consuming and further prolonged by the computational time required for topology

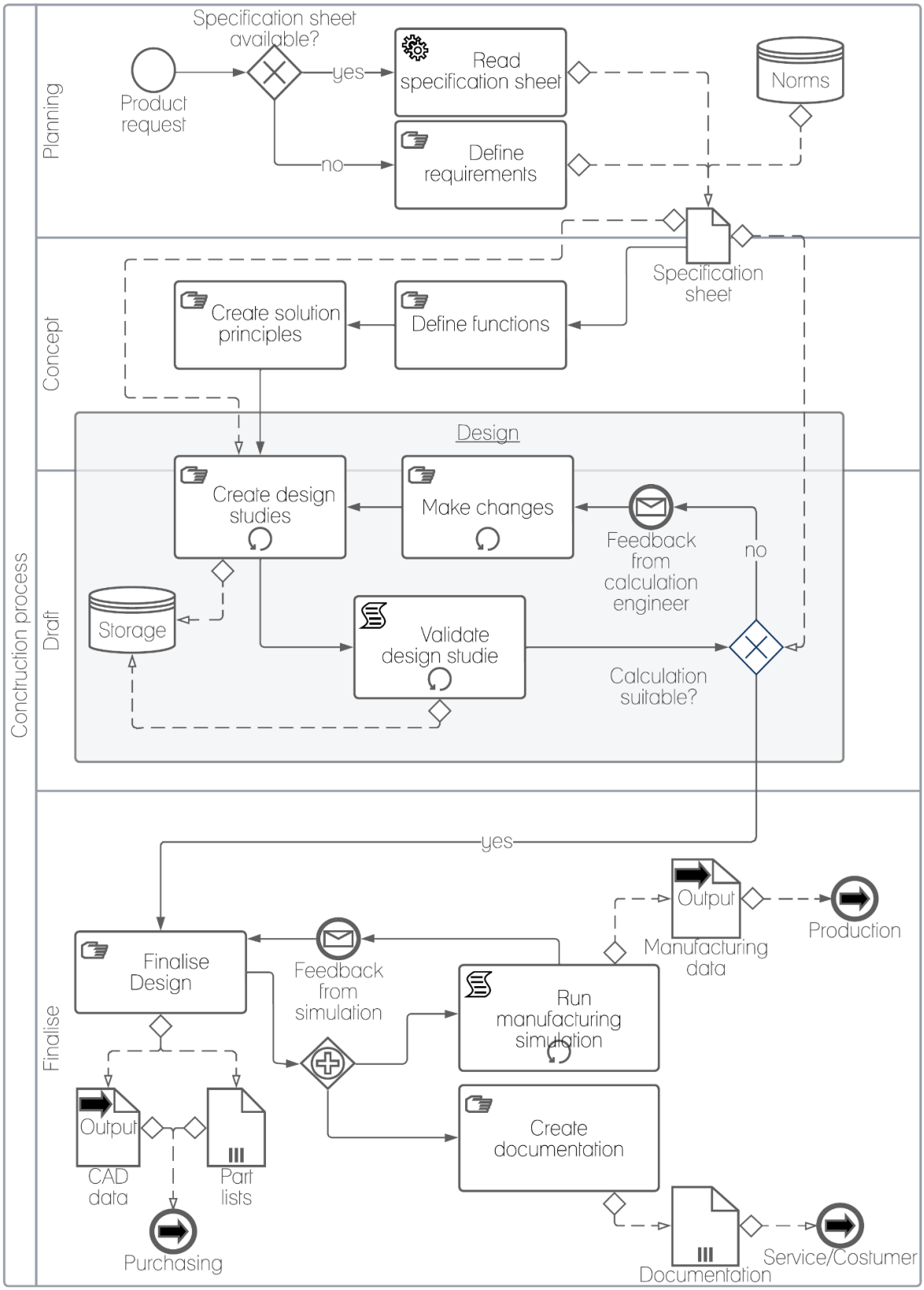

Figure 4: BPMN model of the construction process. The design phase (grey) is identified as a phase for potential time savings and point of application for the proposed method

optimization. However, in early design stages, the high accuracy of traditional topology optimization may not be necessary. Instead, in the context of generative design, the designer can be presented with a potential design based on mechanical calculations without the need for final calculations. This can reduce the iterations in the loop and decrease effort at all levels. Once a valid design is achieved on the mechanical side, minor adjustments for the manufacturing process can be incorporated, which can then be verified through manufacturing simulations. The subsequent process involves forwarding the design to purchasing, production, and documentation.
From the problem analysis, several areas of action have already been identified. One of these areas is the correct and well-founded integration of the method into the development process, which can be considered achieved through the previously described approach. Simple design spaces, minimal preparation effort, and rapid design generation can be effectively incorporated to save time. To ensure a good integration with partially automated reconstruction methods, a voxel-based representation is employed within the method. Voxel representations offer the advantage of being highly robust against errors. They can be easily post-processed, converted using algorithms like Marching Cubes, or transformed into Non-Uniform-rational-B-spline (NURBS) geometries using subdivision techniques, which can then be read by conventional CAD systems [13].
The drawback of high memory requirements is mitigated by reducing the resolution. However, within the context of the method, this reduced resolution can be considered negligible since only design proposals are provided, which are then used by a designer to create the geometry for FEM analysis.

The temporal division into the development process can be seen as the basis for the further development of the methodical approach. The goal is to develop a method that provides a designer with the ability to generate design studies in the context of generative design, based on mechanically meaningful results, in the shortest possible time and with minimal effort and prior knowledge. Simultaneously, a desire for part independence has been pursued, leading to the validation and combination of various approaches. One approach considered is the Generative Design Approach (GDA) as described in [14], which allows users to add specific areas to a pre-existing design within a development environment. In these areas, users can select different design elements to fill the designated spaces. These solution elements can include various manufacturing processes (see Figure 5).

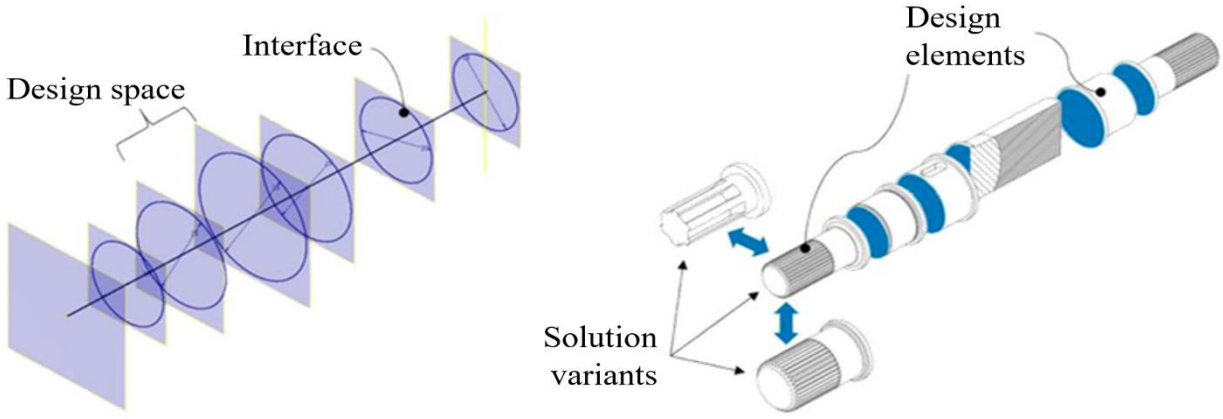

Figure 5: Schematic principle of Generative Design Approach according to [14]

However, the design space is limited in the sense that new design elements need to be added by the user or generated based on rules. The principle of dividing the design into individual design spaces is, nonetheless, a very effective approach for selecting and modifying specific areas of a component. Similarities can be drawn, for example, to the procedures used in finite element analysis. A complex geometry and its properties cannot be precisely determined analytically, as it is not entirely clear how such a geometry will behave under mechanical constraints. However, by dividing this complex geometry into simple, smaller elements, a mechanical analysis can be performed by considering their collective behavior. At the nodes and transition points, information about the neighboring elements must be passed on to analyze their behavior within a component.

In the context of design generation, the topics of topology optimization and generative design have been described above. Topology optimization, due to its dependency on finite element analysis, provides a more precise method but is also associated with longer computation times. Generative design, in its basic definition, can bypass this dependency but may not necessarily deliver mechanically precise results. Therefore, a goal is to reduce the computation time in topology optimization to incorporate the characteristics of generative design, which emphasizes fast result generation. Various possibilities emerge in this regard. On the one hand, computational power can be increased through hardware changes, although a linear relationship is not observed.

Another option is the use of neural networks to generate predictions of optimization results. There are different objectives to be achieved through their implementation. For this method, the aim is to generate designs quickly, thus the goal of a neural network in this context is seen as predicting designs. One approach to achieve three-dimensional predictions from neural networks in the context of topology optimization is to employ Convolutional Autoencoders. Several attempts have been made in the literature, but often limited to two-dimensional cases. According to [15], it is possible to train a network with different initial states of an optimization and use the final structure as the label in the training process. In [15] the authors optimized a cuboid with a fixed design space and various boundary conditions, resulting in a dataset of around 6000 optimized cuboids. However, a problem arises here, as neural networks generally do not possess strong abstraction capabilities, which means that individual components can be predicted only if they are familiar to the network. Nevertheless, the method utilizes the possibility of predicting simple geometric shapes combined with various boundary and force conditions to generate a dataset of a finite number of differently shaped primitives under diverse loads. In order to infer the behavior of an overall component from this dataset, the method must be divided into three fundamental steps, each requiring specific sub-functions to be fulfilled (see Figure 6):

- Initial preparation for training the network - Construction of the dataset:
  - Generate primitives,
  - Apply boundary conditions,
  - First iteration and final iteration for dataset creation,
  - Primitives and FEM constraints as input for training;

- Methodological steps for preparing the optimization of a geometry
  - Capture input consisting of the overall geometry, global forces, and boundary conditions,
  - Segmentation and generation of primitives,
  - Conversion of global load cases to local load cases,
  - Primitives and FEM constraints for the initial iteration of optimization;

- Methodological steps for generating the optimized geometry
  - The first iteration is used to generate predictions of the primitives through the neural network,
  - Subsequently, these primitives need to be assembled into a complete geometry,
  - Initially, the transition areas are simplified by representing them through interfaces, which offer further optimization opportunities.

Method

The ultimate goal is to enable a neural network to predict optimized structures from generic structures and mechanical boundary conditions. In the current state-of-the-art, there exist some restrictions to solve this general task. First, predicting an optimized structure from mechanical boundary conditions, such as force vectors and mechanical bearings directly, is not yet possible because of data sparsity in the input. [11, 16] solve the first iteration of the SIMP algorithm as an intermediary step to enable a neural network to predict final optimized structures with high accuracy for specific design problems. The other restricting factor is the availability of data. In both works, the authors manually create large datasets for their specific design task. Because creating such a dataset involves regular topology optimization it is very time consuming and thus is not a viable approach for generic design tasks.

To enable machine learning (ML) accelerated topology optimization for generic design tasks, the proposed generative design method with ML-assisted topology optimization consists of three complementary steps. Instead of learning to optimize arbitrary complex objects, the design space is segmented into multiple primitive spaces, i.e., cubes, cylinders, etc. This approach is in line with common workflows in CAD and generative design environments such as Constructive Solid Geometry (CSG). The global mechanical boundary conditions are calculated for each primitive by translating forces over the interfaces of each primitive. Fittingly, a dataset is created which consists of parametric primitives optimized for varying mechanical boundary conditions. Here, the SIMP algorithm is applied for the minimization of compliance, although other algorithms and optimizations goals can be applied similarly. The solution of the first iteration is input and the final iteration, where the optimization converges, is the neural network target. A Convolutional Neural Network (CNN) autoencoder is trained on this dataset to predict the optimized structure from the approximated solution of the first iteration. Finally, the ML predicted structures are reassembled according to their interfaces.

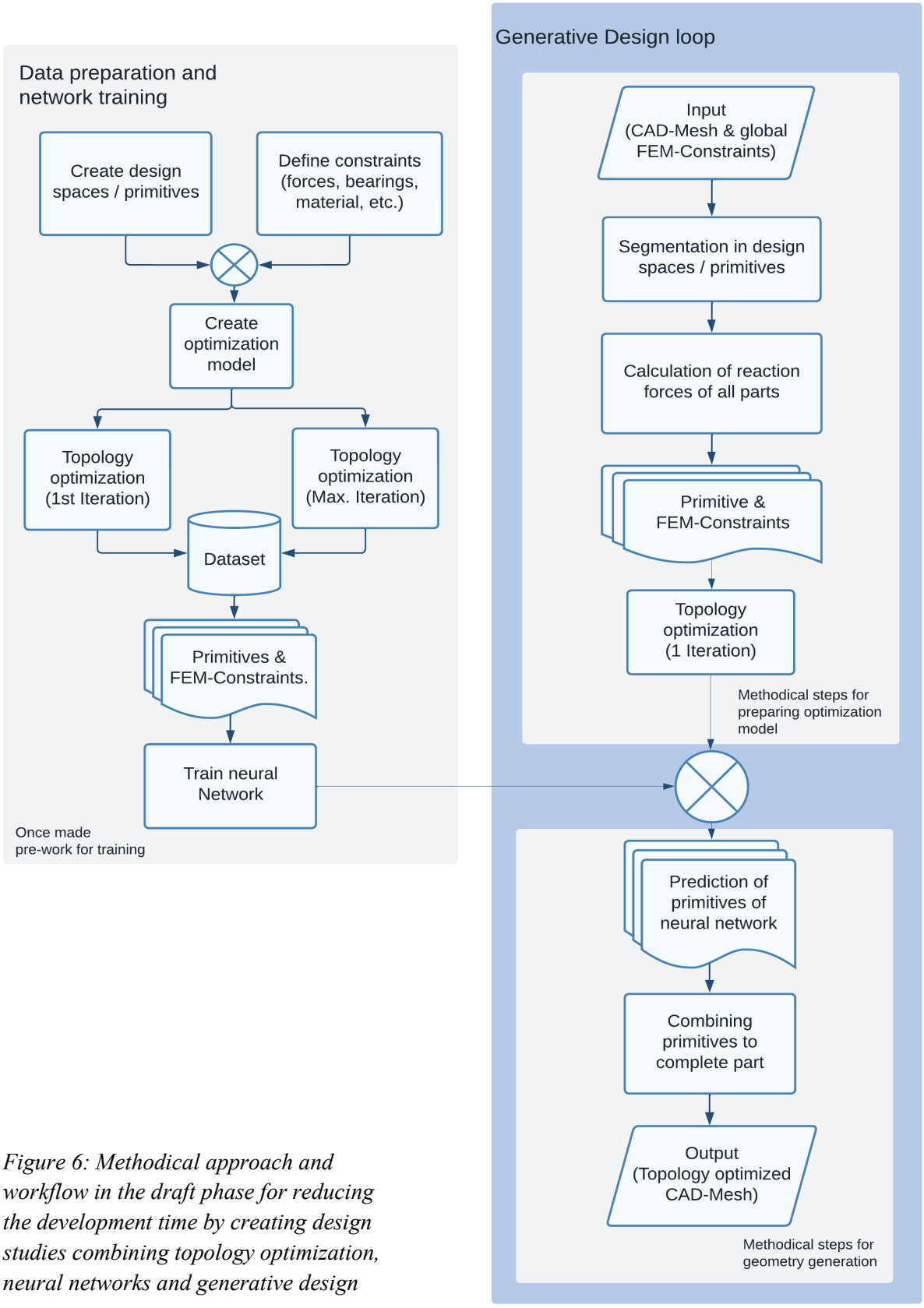

Figure 6: Methodical approach and workflow in the draft phase for reducing the development time by creating design studies combining topology optimization, neural networks and generative design

Proof of concept

Following the implementation of a proof-of-concept is described in detail. We use Synera [S1] as a CAD environment and user interface, Altair Optistruct [S2] to implement part segmentation and topology optimization, PyTorch [S3] for the CNN autoencoder and Weights & Biases [S4] for tracking of the training process and hyperparameter tuning.

Part Segmentation

A mesh or voxelized object is taken as input for the part segmentation. The input is transformed into a centered and normalized voxelization with dimensionality (64, 64, 64). Then, the gradient of cross-sectional area is tracked over every voxel in Z-direction. Additionally, the number of discontiguous voxel spaces, i.e., areas that are not connected to any other active voxels, are recorded. For each separate voxel space, the Z-gradient acts as an indicator for new primitives. If a threshold for the Z-gradient is exceeded, a new primitive is created. Each primitive is represented by a cuboid that includes all of the original's active voxels. Finally, each primitive is rescaled into a (64, 64, 64) voxel space. The object consistently maintains a maximum dimension of 64 units, while the other dimensions undergo perspective-based rescaling in order to match the primitives of the training dataset. In order to guarantee the connectivity of all optimized primitives, the intersecting areas between primitives are non-design-space, i.e., active voxels.
Figure 7 shows the workflow and results for an exemplary part segmented to primitives and the networks input consisting of the first iteration of the single primitives and their reconnection afterwards.

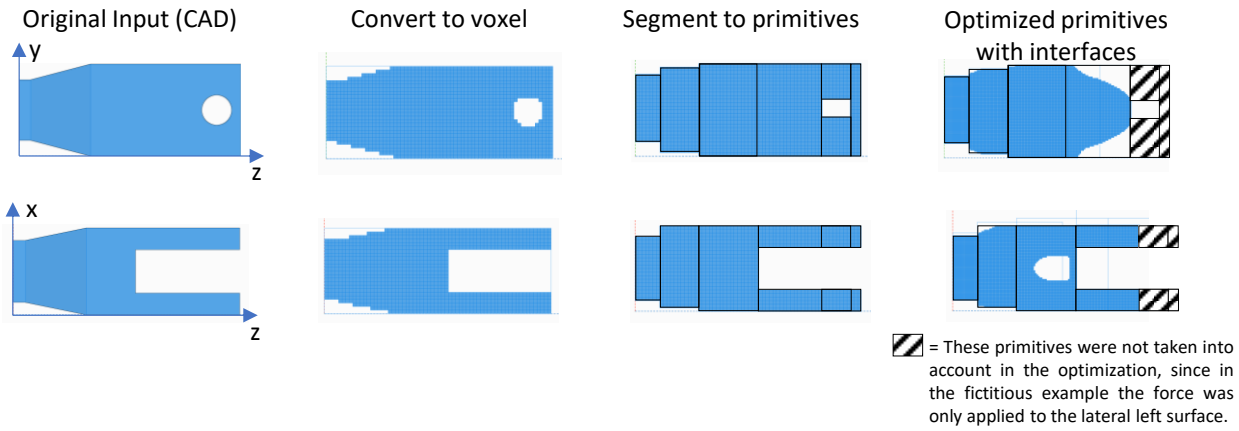

Figure 7: Results and workflow of the segmentation to primitives from complete part

Force Calculation

The user defines global boundary conditions of a traditional topology optimization, such as load vector, point of application, fixed displacement conditions and non-design space. Each primitive is checked for intersection with the points of application of the external force. The resulting forces in each fixed displacement condition are calculated. The resulting forces with inverse vector direction are applied to the vertices of neighboring primitives and their resulting forces are calculated. This process is repeated iteratively until internal forces of all primitives are calculated.

Dataset Generation

The dataset is generated so that the boundary conditions match possible results of the part segmentation. Each primitive within the dataset is placed in a (64,64,64) voxel space. The dimensions of the actual design space are selected randomly as multiples of 8, where one dimension is always 64.

Loads and mechanical bearings are applied randomly on the lowest and highest voxel rows in order to match the segmentation in Z-direction. In particular, one to four rectangular area loads and mechanical bearings are applied in random positions to each primitive. The load vector F is selected randomly as $F(F_x, F_y, F_z)$ with $F_x,\ F_y,\ F_z \in \{-1000, 1000\}$.

For topology optimization the SIMP algorithm is applied to minimize compliance. The minimum weight is restricted to 0.25. The ISO value is set to 0.14. The optimized mesh is converted into a voxel space as described above. Input and target for the neural network are the first and last iteration of the SIMP algorithm. We generate around 8000 samples, exemplary depicted in Figure 8.

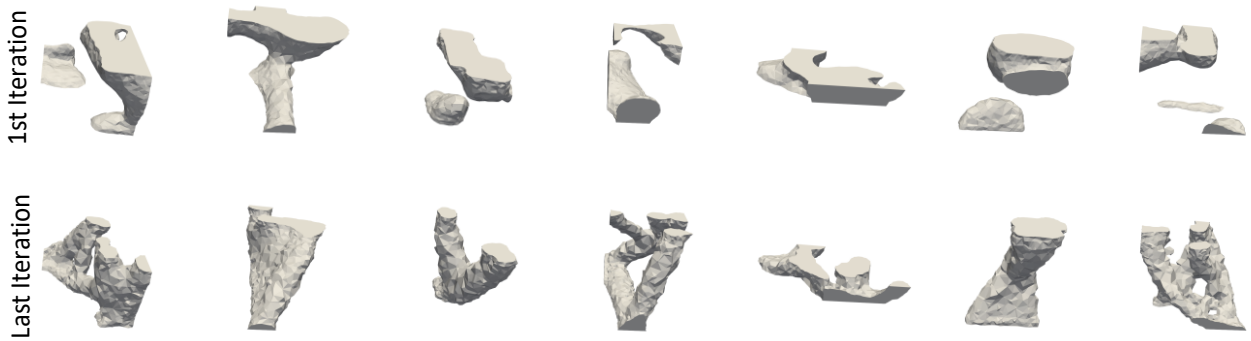

Figure 8: Exemplary structures within the dataset for primitive's optimization

Neural Network

The CNN autoencoder architecture is represented by a 3D U-Net [15]. In the encoder several convolutional layers reduce the spatial dimensionality of the input geometry. Instead of pooling layers, we employ learnable convolution layers with stride 2. The decoder essentially mirrors the decoding steps of the encoder by employing transposed convolution layers. Padding is applied, in order to control the spatial dimensions of the feature maps and match them where skip connections are applied. Two skip connections enable earlier feature maps, to bypass the informational bottleneck created by the encoder. The respective feature maps are concatenated with those of later layers. Employing skip connections allows the model to reconstruct fine grained details in the output prediction. The model aims to predict whether or not there is material distribution in a voxel. For this binary decision, we employ Binary Cross Entropy as loss function and a Sigmoid activation function in the last layer. Thus, we apply Categorical Crossentropy loss and Softmax activation in the last layer. L2 penalization is added to the loss term as a means of regularization. The models are trained using gradient descent with Adam optimizer for 5000 epochs. The architecture achieved an accuracy of 99.9% across all voxels with the settings made and the data set described, however the number of correctly predicted active voxels chosen as a more meaningful metric. The network achieves a performance of 78% prediction accuracy.

Figure 9 first shows a possible result of a neural network prediction based on the input of the first iteration of the primitives. The voxel representation of the networks input is shown completely on the left side. The middle view shows the prediction of the network and the highlighted area on the right side of the figure shows here once the structures via a conversion as STL representation, because of the limited depth representation of the voxels. It can be seen that the neural network is able to predict the individual primitives in an optimized form and uses structure-optimized patterns for this purpose, such as the visible supports of the upcoming interface on the right view.

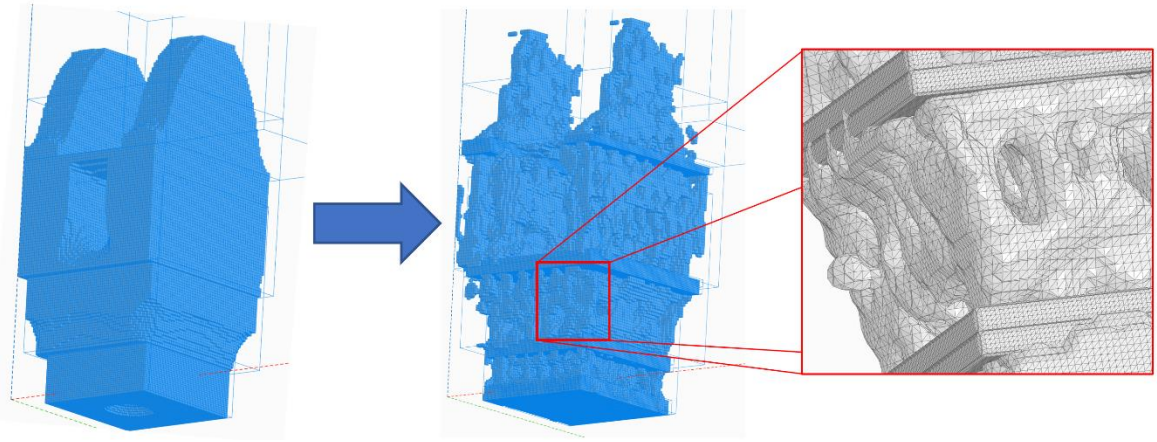

Figure 9: Left: first iteration input to neural network; middle: prediction of optimized primitives of neural network; right: detailed view for optimized structures in STL representation

Summary and Outlook

The proposed method enables near real-time topology optimization in a generative design environment by application of machine learning. The proof-of-concept is fully embedded within a CAD environment via Synera and allows different input and output formats. A basic user interface is provided for defining boundary conditions such as load vector, mechanical bearings and non-design space.

The method aids in reducing time spent in early stages of product development by using neural networks. For this purpose, we identified key stages in the design process that allow for increased efficiency based on further automation. Specifically, this method improves the iterative process between creating design studies, evaluation and making changes based on mechanical properties. Minimizing iterations between design and simulation reduces friction and leads to shorter development time. The relevance of these issues was verified by regular consultations with project partners within BIKINI project named under acknowledgments.

The outlook can be best determined by referring back to the initial requirements set in chapter [Problems and need of action]. This enables an assessment of the current capabilities of the method and its implementation, as well as identification of any existing methodological or technical limitations:

Low level of user knowledge for broad application:
Using this method does now require more knowledge than a traditional topology optimization. The user defines global mechanical boundary conditions, such as loads, mechanical bearings and design

space. Input and output support all conventional CAD formats. Segmentation into primitives, calculating internal mechanical conditions and ML assisted topology optimization run automatically within the method.

Consideration of mechanical boundary conditions and designs:
The proposed method considers global mechanical boundary conditions and calculates its impact on each primitive. By applying topology optimization to each primitive instead of the exact input, there exist some inaccuracy compared to a traditional approach. This is due to varying design space and conversion of boundary conditions. This method does not replace traditional topology optimization, but instead creates new application in a generative design context.
Furthermore, there exist some limitations in the current state-of-the-art of ML application in topology optimization. Training a CNN on a voxelized 3D object is computationally expensive and time-consuming in the training process, thus limiting the resolution of the input and output of the neural network. Current solutions cannot predict an optimized structure from numerical boundary conditions and expect an early iteration solution of traditional topology optimization.

Effortless customization:
A user enters input conditions such as geometry and mechanical boundary conditions. Every other step in the method is fully automated, returning a topology optimized geometry based on the users' inputs.
Since the decomposition into individual subproblems abstracts the global problem, the overall construction of the dataset, the algorithm for decomposition into primitives, as well as the conversion of forces and constraints, only need to be implemented once. They can then be applied to various use cases or modifications without incurring additional effort. Depending on the final use case, the process can be adjusted, e.g., adding primitives, changing the topology optimization algorithm, etc.

Generic and component-independent application:
It has been elucidated how ML assisted topology optimization can be applied in a component-independent manner. Some of the functions already fulfill this purpose, i.e., the decomposition into primitives, the conversion of all conditions, the construction of the resulting subproblems, and the conventional topology optimization of these subproblems, followed by their assembly through interfaces. The advantage of the method lies in its ability to abstract complex problems and thus achieve generalizations.
The generation of training data needs to be tailored to the specific primitives and mechanical boundary conditions existent in the ground-truth object. Using a different part segmentation algorithm or adding primitive types would necessitate changes to the training dataset. The design space of generated data samples is defined by the properties listed in chapter [Proof of Concept – Dataset Generation]. Currently, there exists only one type of primitive, i.e., cuboids. Adding different primitives is possible, but requires more training data. The part segmentation algorithm is rudimentary and currently only allows segmentation in one direction. There exist ML applications [16] that could be applied to help solving this problem in the future.

Time reduction:
Time savings are achieved primarily through a reduction in iterations between design and analysis. Topology optimizations of components typically take several hours, and the more frequently this step needs to be repeated due to adjustments, the stronger the benefit of directly predicting

optimized structures becomes. By utilizing neural networks to reduce the actual calculation time of the optimization to a few seconds, significant hours can be saved in product development, thereby reducing both the time-to-market and development costs.
In the proof of concept presented here, a calculation time of around 17 minutes was achieved for the calculation using conventional topology optimization, whereas the overall workflow had an overall runtime of only 4 minutes. This corresponds to a time reduction of 77% which makes this process more viable in a generative design context. In further validation and test procedures, these times will also be further investigated on other use cases.
Creating a large dataset consisting of topology optimized primitives and training the neural network requires preliminary work. Creating the dataset takes around seven days and training the neural network about 36 hours. Nevertheless, this work only needs to be done once resulting in near-instant, component-independent topology optimization in application.

Acknowledgments

This work is a part of the research work from the project "Bionics and AI for Sustainable Integration in Product Development for Resource Efficient Lightweight Design" (funding code 03LB3018C), which is funded by the Federal Ministry for Economic Affairs and Climate Action. Some information is based on interviews and therefore information from project partners, which do not have an explicit reference though.
The machine learning models were trained on Nvidia A100 80GB GPU in combination with two Intel Xeon Gold 6346.

References

[1] G. Pahl, W. Beitz, J. Feldhusen et al., *Konstruktionslehre*: *Grundlagen erfolgreicher Produktentwicklung ; Methoden und Anwendung*, Springer, Berlin, Heidelberg, 2007.
[2] S. Vajna, C. Weber, K. Zeman et al., *CAx für Ingenieure*: *Eine praxisbezogene Einführung*, Springer Vieweg, Berlin, Germany, 2018.
[3] A. Nordin, "Challenges in the industrial implementation of generative design systems: An exploratory study," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, vol. 32, no. 1, pp. 16–31, 2018.
[4] F.P. Brooks, *The design of design: Essays from a computer scientist*, Addison-Wesley, Epper Saddle River, 2010.
[5] D. C. Wynn and C. M. Eckert, "Perspectives on iteration in design and development," *Research in Engineering Design*, vol. 28, no. 2, pp. 153–184, 2017.
[6] H. Wögerbauer, *Die Technik des Konstruierens*, Oldenbourg, München, 1942.
[7] L. Harzheim, *Strukturoptimierung*: *Grundlagen und Anwendungen*, Verlag Europa-Lehrmittel Nourney Vollmer GmbH & Co. KG, Haan-Gruiten, 2019.
[8] C. Hessel, *Integration der Topologieoptimierung in den CAD-gestützten Entwicklungsprozess*, Shaker, Aachen, 2003.
[9] *Design for life*: *The 20th International Conference on Engineering Design (ICED 15); 27th - 30th July 2015, Politecnico di Milano, Italy*, Design Society, Glasgow, 2015.
[10] M. Ott, N. Meihöfener, and R. Koch, "Neuronale Netze in der Konstruktion zur Ausschöpfung der Potentiale additiver Fertigungstechnologien," in *Tagungsband des DVM-Arbeitskreises "Additiv gefertigte Bauteile und Strukturen"*, vol. 7, pp. 91–106.
[11] J. Freund and B. Rücker, *Praxishandbuch BPMN*: *Mit Einführung in DMN*, Hanser, München, 2019.

[12] T. Reiher and R. Koch, “FE-Optimization and data handling for Additive Manufacturing of structural parts,” in *Solid Freeform Fabrication Symposium*.
[13] T. Reiher, *Intelligente Optimierung von Produktgeometrien für die additive Fertigung,* Dissertation, Shaker Verlag.
[14] K. Hermann, O. Altun, W. Philipp et al., “Methodischer Aufbau von Entwicklungsumgebungen nach dem Generative Parametric Design Approach,” *32. DfX-Symposium*, 2021.
[15] S. Banga, H. Gehani, S. Bhilare et al., “3D Topology Optimization using Convolutional Neural Networks,” 8/22/2018, https://arxiv.org/pdf/1808.07440.
[16] K. Yang and X. Chen, “Unsupervised learning for cuboid shape abstraction via joint segmentation from point clouds,” *ACM Transactions on Graphics*, vol. 40, no. 4, pp. 1–11, 2021.

Software references

[S1] Synera GmbH: Synera [online], https://de.synera.io/. (Accessed 29.06.2023).
[S2] Altair Engineering: Altair Optistruct [online], Available at: https://www.altair.de/optistruct/. (Accessed 29.06.2023).
[S3] PyTorch: PyTorch - An open source deep learning platform [online]. Available at: https://pytorch.org. (Accessed 29.06.2023).
[S4] L. Biewald: Experiment Tracking with Weights and Biases [online], Available at: https://wandb.com/. (Accessed 29.06.2023).