***Proceedings of the IASS 2024 Symposium***
*Redefining the Art of Structural Design*
August 26-30, 2024, Zurich Switzerland
Philippe Block, Giulia Boller, Catherine DeWolf,
Jacqueline Pauli, Walter Kaufmann (eds.)

# Structural Embodiment – Unified Workflow and Toolkit for Form-finding, Solid Geometry Generation and Visualisation via Deep Learning Methods

Tao Sun*, Pierluigi D'Acunto[a b], Frank Petzold[c d]

* Professorship of Structural Design, Technical University of Munich
Arcisstr. 21, 80333 Munich
tao.sun@tum.de

[a] Professorship of Structural Design, Technical University of Munich
[b] Institute for Advanced Study, Technical University of Munich
[c] Chair of Architectural Informatics, Technical University of Munich
[d] Munich Data Science Institute, Technical University of Munich

## Abstract

This paper introduces a deep learning-driven workflow and toolkit, the Structural Embodiment Toolkit on the McNeel Grasshopper platform, that supports the conceptual structural design process by streamlining form-finding, solid geometry generation, and AI-powered visualisation within a single CAD environment. The toolkit equips users with adaptors for incorporating results from established form-finding tools such as Combinatorial Equilibrium Modelling (CEM) and Kangaroo Physics, facilitating the transformation of the structural skeleton into solid geometries with appropriate cross-section based on internal forces. Leveraging the transformative potential of deep learning, it allows user-friendly access to deep learning-based image generation within Grasshopper with concurrent multiple control options, including line, depth and semantic segmentation. The toolkit also offers components to enhance the conventional render-based visualisation pipeline with the ability to mass-produce for model training, embodying a self-reinforcing ecosystem. This streamlined process and toolkit, exemplified through a detailed case study to demonstrate improvements in design ideation, communication, and collaboration, underscoring the distinctive potential of deep learning in structural design.

**Keywords**: Deep learning, graphic statics, parametric design, form-finding, solid geometry generation, visualisation, stable diffusion, digital design tool

## 1. Introduction

Deep learning (DL), a subset of machine learning (ML) that employs neural networks, serves as a cornerstone for modern artificial intelligence (AI) advancements, encompassing areas such as speech processing, computer vision, and natural language processing [1]. Recently, DL applications have begun transforming daily life by significantly enhancing efficiency across various sectors. DL has been instrumental in improving decision-making, productivity, and overall system performance in industries like manufacturing, contributing to greater sustainability [2].

In parallel, the Architecture, Engineering, and Construction (AEC) industry is also capitalising on DL's potential. Recent attempts include leveraging DL for tasks such as interpreting architects' hand drawings to predict design phases [3], using Graph Attention Networks (GATs) for the detection and analysis of geometric primitives in CAD drawings [4], and employing Recurrent Neural Networks (RNNs) to

suggest next steps in early design stages by learning cognitive sequences in Building Information Modeling (BIM) [3]. Specifically, the structural design field is actively exploring DL's capabilities.

### 1.1 Structural Design via Form-finding

Structural design is the process of creating physical structures that can withstand external forces and meet functional requirements. Form-finding is one of the structural design processes that lead to the definition of the form of a structure in static equilibrium under given loads and boundary conditions, resulting in structurally efficient structures that promote more sustainable use of material. It could be both physical and digital.

#### *1.1.1 Historical Development*

Early pioneers like Antoni Gaudi utilised funicular structures, notably in Sagrada Familia, using rope models to simulate tensile forces. Frei Otto's experiments with soap films, Heinz Isler's development of shell structures using hanging cloths, and Sergio Musmeci's experiments with mathematical models further advanced form-finding methodologies and paved the way for the development of computational form-finding [5].

Recent developments in digital form-finding, like the introduction of Kangaroo Physics [6], RhinoVault [7], EQlib [8] and Kiwi!3D [9], have enriched the field of computational structural design. Commercial software such as SOFiSTiK [10], Easy software [11], and the Oasys GSA Suite [12] now feature Rhinoceros/Grasshopper interfaces, highlighting the continued integration of computational tools in design workflows. In the context of computational form-finding with graphic statics, D'Acunto et al. (2019) [13], and Ohlbrock et al. (2020) [14] contributed with a Vector-based 3D Graphic Statics framework and Combinatorial Equilibrium Modelling (CEM), respectively. In 2021, Shen et al. introduced the Vector-based Graphic Statics (VGS) tool for spatial structure design within the Rhino/Grasshopper environment [15].

#### *1.1.2 Form-finding Enhanced by Machine Learning*

The integration of ML with structural design has evolved through collaborative advancements, each contributing uniquely to the field's development. In 2018, Fuhrimann et al. explored machine learning for structural form-finding, utilising spatial networks to broaden the understanding of solution spaces [16]. This was furthered by de Miguel et al. (2019), who applied variational autoencoders for generating structural typologies, albeit without actual form-finding, pointing to areas needing enhancement [17]. In 2020, Bertagna et al. broadened the scope to include rapid exploration of topological designs, particularly for compression-only shell structures, yet left solid geometry generation and visualisation processes unaddressed [18]. In 2021, Ochoa et al. present a CAD framework that leverages human and machine intelligence to generate and evaluate non-standard structural forms in static equilibrium, using CEM, self-organising maps, and gradient-boosted trees for iterative design improvement [19]. In 2022, efforts by Guo et al. and Bleker et al. introduced novel approaches: Guo's team combined natural language processing with graphic statics for form-finding [20], while Bleker automated graph modelling using GNNs [21], pushing the boundaries further in structural design through ML.

#### *1.1.3 From form-finding to visualisation*

Conceptual structural design goes beyond creating a structural skeleton; it demands accurate representation throughout its process. The traditional workflow consists of three primary steps: form-finding, solid geometry generation, and render-based visualisation (Figure 1). Following form-finding, the dimensions of structural members are often undefined, necessitating a next step to transform the skeleton into tangible solid geometry. Subsequently, designers manually build detailed site models and spend considerable time fine-tuning materials and lighting for visualisation purposes. Image-generating AI has emerged as a potential solution to these labour-intensive steps.

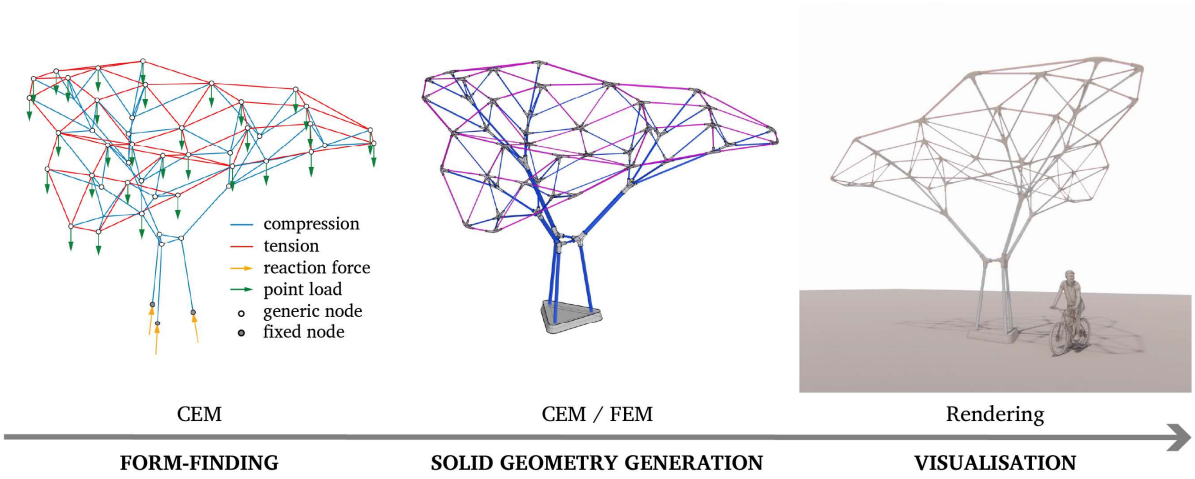



Figure 1: Conventional render-based workflow exemplified via a canopy design form-found by CEM

## 1.2 Image Generating AI as an Alternative for Rendering

Introduced by OpenAI in 2022, ChatGPT [22] highlighted AI's capabilities in language processing, pointing the direction with its large language models (LLMs) for text-to-image models by understanding human languages in multi-dimensional vector spaces [23]. DALL-E [24], integrating diffusion models with Contrastive Language-Image Pretraining's (CLIP) embeddings for image and text [25], showcased advanced image generation, albeit with limitations in customisation and ways of access. Midjourney [26] and Stable Diffusion [27] represent the evolution of image generation via diffusion models. Stable Diffusion stands out for its local deployment capabilities on consumer-grade GPUs, introducing high-quality, highly customisable image generation.

### *1.2.1 Control, Customizability and Speed*

ControlNet enhances diffusion models by adding precise guidance with inputs like sketches, semantic segmentation or depth maps [28], which is crucial for visualising structural designs with high geometrical accuracy. However, controlled GAN-based methods [29] and architectural visualisation plugins like ArkoAI [30] and Veras [31] offer limited utility due to their unripe development or cloud-based limitations. Customisation and speed are vital, with Stable Diffusion models meeting these needs to some extent. Yet, their lack of domain specificity led to the development of specialised models through methods like fine-tuning. Low-Rank Adaptation (LoRA) emerged as a viable alternative to customise models by reducing trainable parameters [32] and becoming more cost and time-efficient as the base models' sizes are growing significantly. For speed, latent consistency models (LCMs) streamline the image generation process, offering integration as additional LoRA models for easier access to substantially faster generation [33].

### *1.2.2 Stable Diffusion Access*

Scriptively accessing Stable Diffusion ranges from modifying the original codebase to ready-to-use scripts like Hugging Face's pipeline. Graphically, ComfyUI [34] offers a node-based user interface (UI) for Stable Diffusion. The Stable Diffusion WebUI [35] owns the most prominent user and developer community. The Ambrosinus toolkit [36] for Grasshopper functions on the WebUI's API showcases the potential for CAD integration despite current limitations in user-friendliness and the lack of support for advanced features such as concurrent multiple ControlNet units. These developments highlight the need for further investigation and enhancement in integrating Stable Diffusion into architectural workflows.

## 1.3 Motivation

The conventional structural design workflow in the conceptual phase is limited by the inefficiency of frequent tool switching and a lack of integration between form-finding, solid geometry generation, and visualisation. This fragmented approach not only prolongs the design process but also diminishes productivity and creativity. Despite the abundance of digital tools, especially in Rhino and Grasshopper

environments, there's a noticeable gap in creating a seamless workflow that minimises manual labour. The advent of AI-based visualisation, particularly through controlled text-to-image models, presents a promising avenue to address these challenges. By streamlining the design process within a unified CAD environment and harnessing the power of AI for efficient visualisation, there's potential to significantly enhance both the speed and quality of conceptual design, reducing time-consuming tasks and fostering better design communication.

## 2. DL-Enabled Workflow and Toolkit

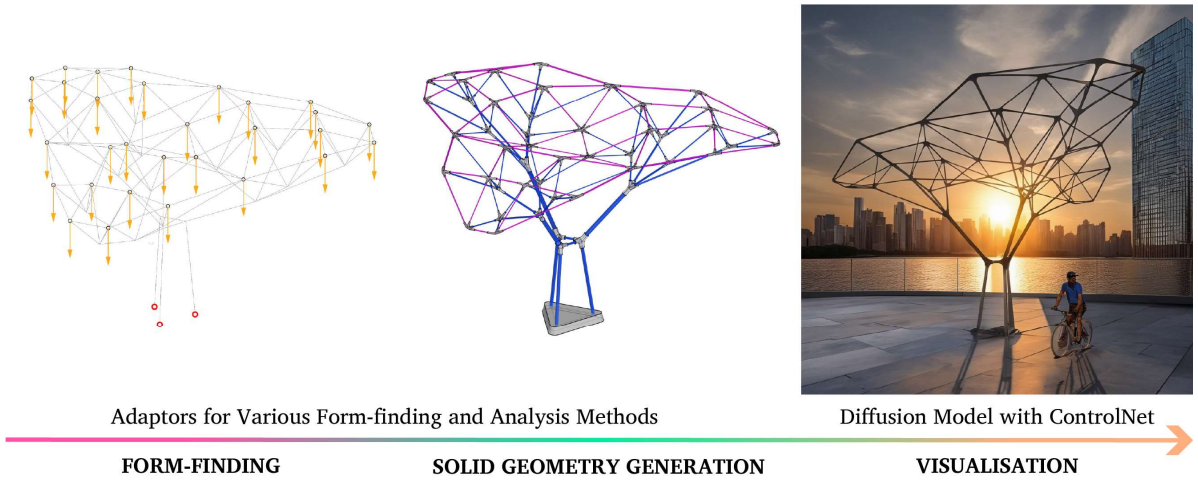



Figure 2: The proposed DL-enabled workflow intended to be established via the SE Toolkit

Addressing the research gaps identified in Section 1.3, this paper introduces a workflow aimed at enhancing the efficiency of the conceptual structural design phase (Figure 2). Compared to the conventional workflow described in Section 1.1.3, this proposed workflow seeks to streamline the processes of form-finding, solid geometry generation, and AI visualisation within a single CAD environment.

The Structural Embodiment Toolkit underpins this workflow by providing components in Grasshopper designed for modularity, reusability, and robustness to fit seamlessly into various design scenarios. It includes adaptors for integrating data from multiple form-finding techniques and tools, generates a solid geometry according to the structural analysis, and builds simple site models. For visualisation, the toolkit leverages stable diffusion with concurrent multiple ControlNet units for greater geometrical fidelity. In addition to that, it also offers components to enhance conventional render-based workflows and enables the mass generation of training data, ensuring a smooth transition between traditional and innovative design methodologies.

### 2.2 Toolkit Development Principles

The Structural Embodiment Toolkit was developed in C# within the Visual Studio IDE using McNeel's official SDK 8.0.23304.9001 [37]; as a product, it is available on the Package Manager of Rhino 8. Aimed at open-source distribution to foster community contribution, the codebase is hosted on GitHub [38], highlighting version control and incremental development. The toolkit also addresses Grasshopper's single-threaded limitation by advocating for asynchronous component execution, enhancing usability during computationally intensive tasks such as text-to-image generation.

#### *2.2.1 Software Architecture*

Structured around object-oriented programming principles, the toolkit's software architecture is divided into three main directories—*Core*, *Components*, and *Properties*—to ensure extendability and ease of maintenance. The *Core* directory forms the toolkit's foundation, containing classes and functionalities designed to support future features. The *Components* directory bridges Grasshopper with the toolkit's *Core*, facilitating user-friendly access to its capabilities from the Grasshopper canvas. The *Properties* directory holds project information and artwork, including components' icons.

### *2.2.2 GUI Design*

The GUI design is based on simplicity, intuitiveness, and a distinctive aesthetic to visually distinguish different design phases and enhance the interface's self-explanatory nature. Custom button interactions deviate from Grasshopper's standard UI, introducing an intuitive user interaction method. (Figure 3).

## 2.3 Component Overview

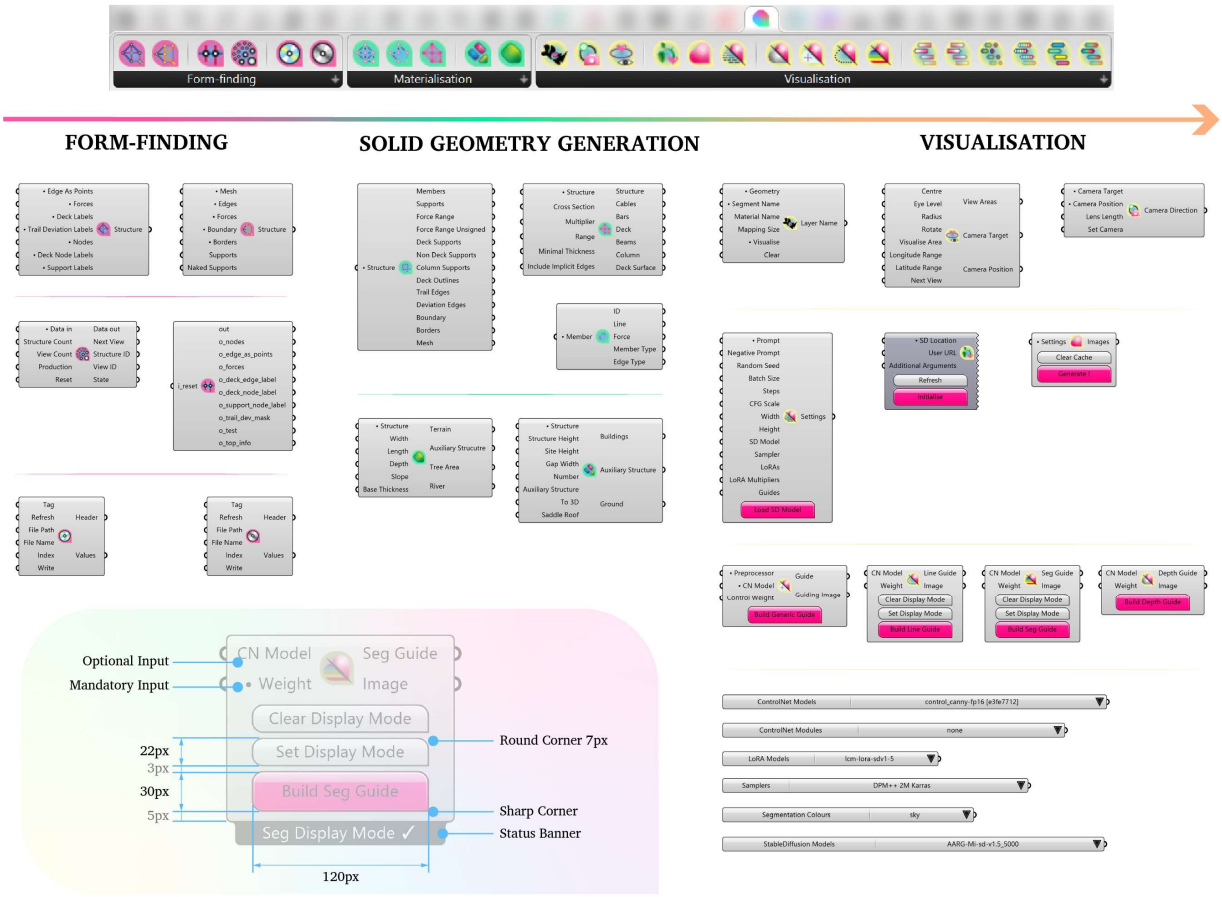



Figure 3: An overview of the SE Toolkit and its GUI design guideline

## 4. Case Study

The Structural Embodiment Toolkit, featuring form-finding adaptors for CEM, demonstrates its capabilities through a case study, which utilises a Python script based on CEM for generating a dataset of 50-meter-long bridges, categorised into arch and suspension bridges, each with distinct design configurations.

### 4.1 Form-finding: CEM Adaptor

Using Rhino 8's *Grasshopper Script component*, the bridge-generating component applies the CEM method to produce outputs like nodes, edges, forces, and specific bridge attributes. The *CEM Adaptor* processes this data, creating a structured object integrated within the SE Core for further operations (Figure 4).

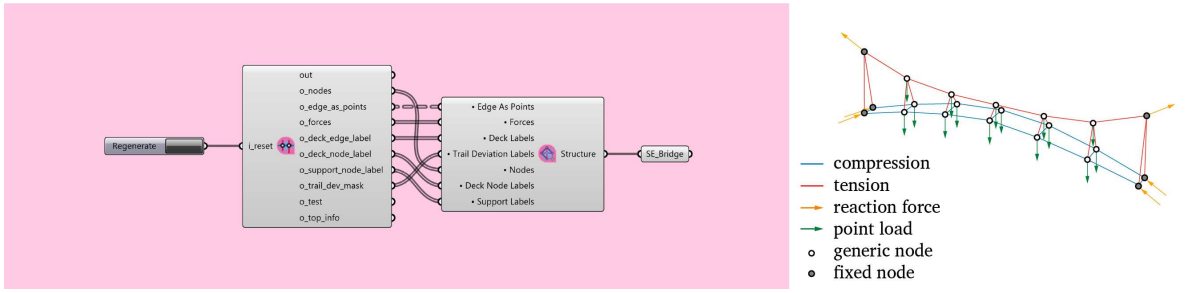



Figure 4: The form-finding step with the CEM Adaptor

### 4.2 Solid geometry generation: Structure, Site, Materials and Texture Mapping

CEM's form-finding specifies force magnitudes essential for the *Materialiser* component to determine member dimensions proportionally, offering users options for cross-section shapes. Output geometries are organised by function for subsequent phases. The *Terrain Maker* component further allows for the creation of parametric site models (Figure 5).

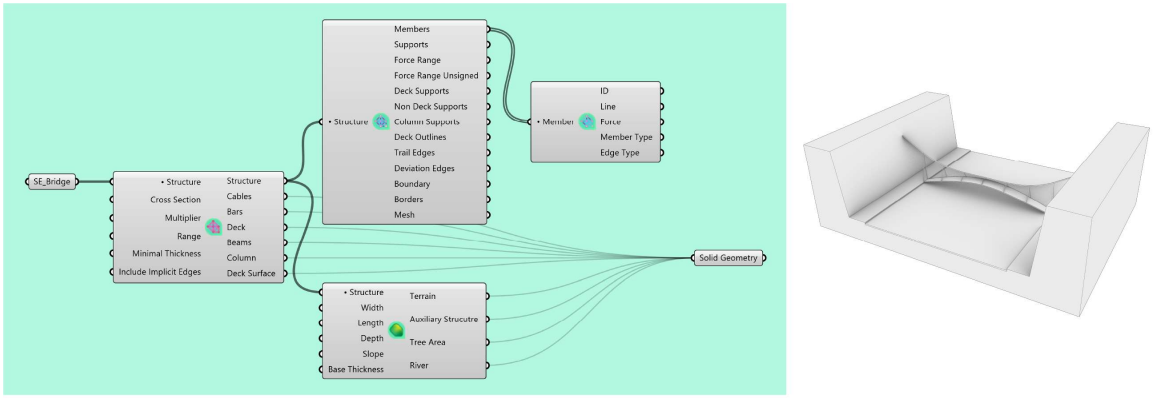



Figure 5: The solid geometry generation step with automatic site model generation

### 4.3 Render-based visualisation: Enhancing the conventional method

In this step, solid geometries are inputted into the *Rhino Visualizer* component for texture mapping and automatic layer organisation, with the *View Randomizer* Component generating camera positions for varied viewing angles. These components then integrate with V-Ray [39] for rendering, demonstrating the toolkit's capacity to enhance conventional visualisation methods with improved efficiency and flexibility (Figure 6).

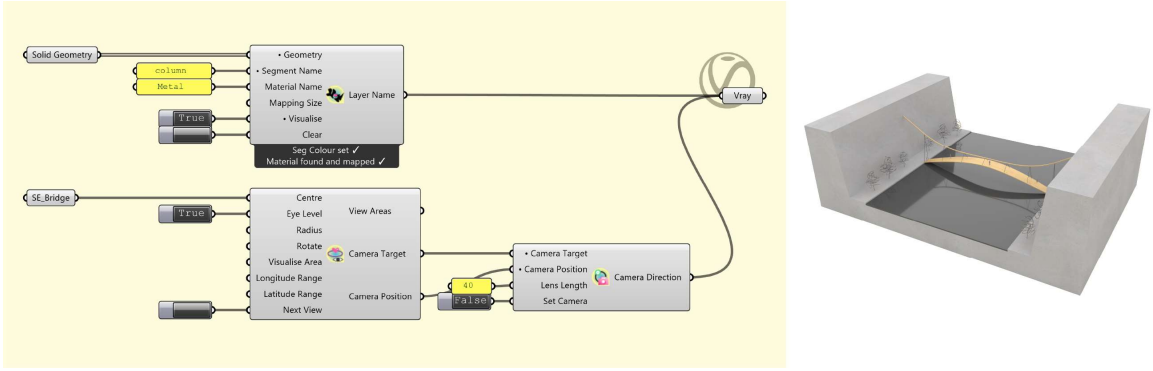



Figure 6: The visualisation step with a conventional render-based solution

### 4.4 Mass-production and Model Training: Self-reinforcing Ecosystem

The SE Toolkit transcends mere compatibility with existing deep learning models by facilitating the mass generation and data recording. This feature is particularly beneficial for creating datasets to train new deep-learning models. The *Mass Producer* component automates the production of designs to specified quantities and views. The *Data Recorder* captures input and output data, allowing for CSV or JSON format saving (Figure 7 left). Using the toolkit, 10 different bridges with 10 views each were generated, creating 100 images of 1024x1024 resolution. These images, categorised into arch and suspension bridges, form a varied but balanced dataset. Rendering these images required, on average, 20.3 seconds each on a machine with a 10 Core Intel i9-10900k CPU. Captioning was manually done to ensure accuracy in the description.

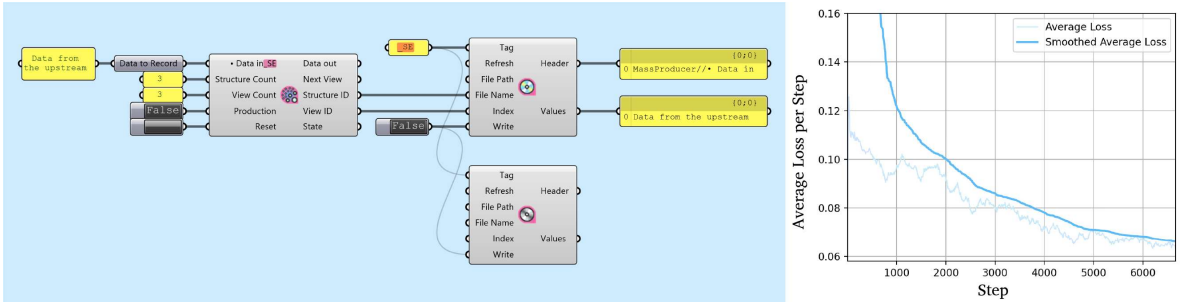



Figure 7: The mass production enabled by the SE Toolkit (left) and the training loss (right)

The training of the LoRA Model was conducted using the Kohya_ss [40] training pipeline, anchored on the runwayml/stable-diffusion-v1-5 [41] source model. This model typically outputs images with a resolution of 512×512 but can be higher resolution. No image preprocessing was required for our purposes, as our objective was to generate images at a resolution of 1024×1024. The training process utilised an NVIDIA RTX 6000 Ada GPU, facilitating 6 batches during training. The training regimen spanned 20 epochs. Each image undergoes sampling 20 times, culminating in 2000 steps. Consequently, the total number of training steps amounted to 6667, calculated as 2000 steps / 6 batches × 20 epochs. The training took 3 hours and 57 minutes, resulting in 20 LoRA checkpoints from each epoch.

The training process demonstrated promising results, with a steady decline in loss that began to plateau over time (Figure 7 right). An XY plot tested weights from 0.2 to 1.0 across 20 checkpoints to find the optimal model and weight setup, revealing that a weight close to 0.8 from checkpoint 16 onwards produced reasonable images using a consistent prompt *"physical model, arch bridge, concrete, site, timber, wire trees, dark acrylic, reflection, studio lighting "* and the same random seed (*2900010133*) (Figure 8). The LoRA models SE_CEM_BRIDGE are available on Hugging Face [42].

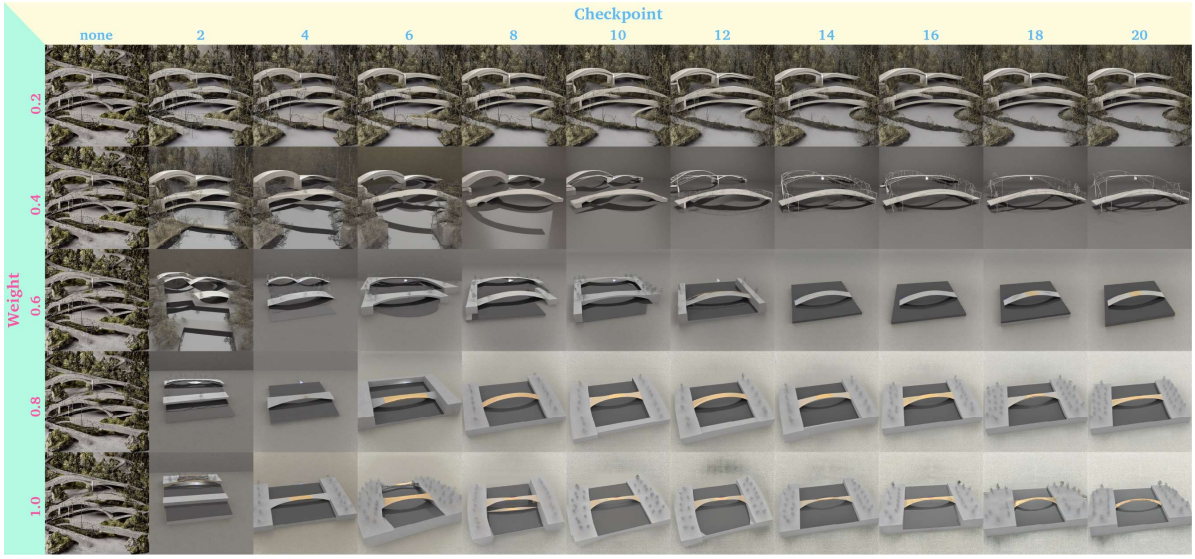



Figure 8: XY-Plot for every second LoRA checkpoint with different weights

### 4.5 From Rendering to AI Visualisation

To assess the LoRA model's generalisation capabilities, two new bridge structures were generated, form-found, materialised, and visualised with the toolkit. Comparing V-Ray-rendered images to AI visualisations revealed a significant reduction in time, from 21 seconds to 7 seconds for the suspension bridge and from 19 seconds to 8 seconds for the arch bridge via the SE Toolkit (Figure 9), using consistent prompt with concurrent depth and canny ControlNet units (depth's weight at 0.6, canny's weight at 0.3, and LoRA's weight at 0.8).

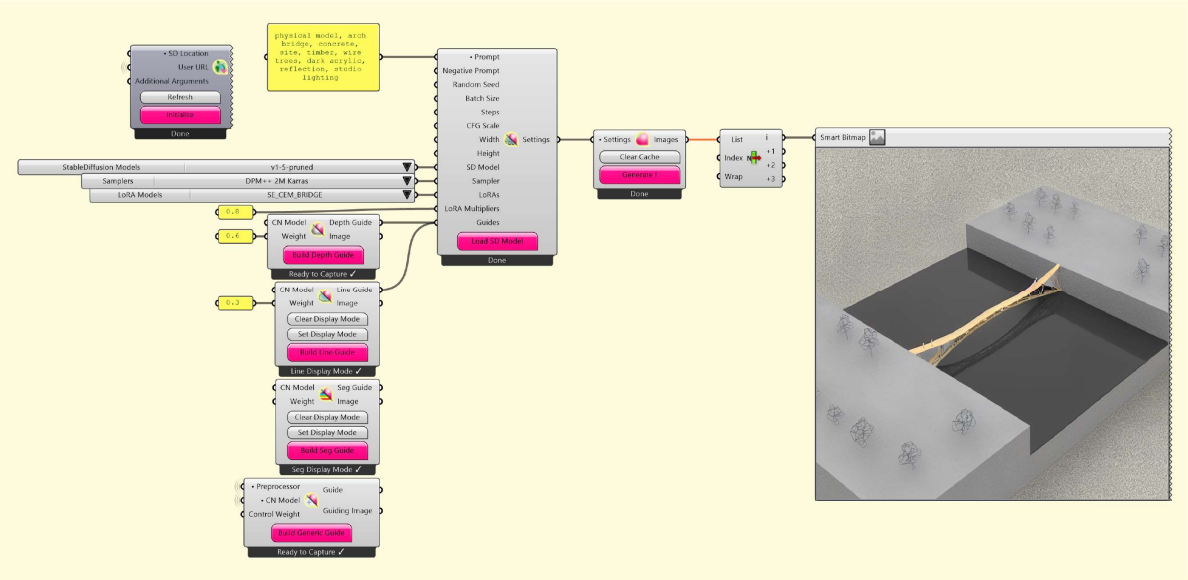
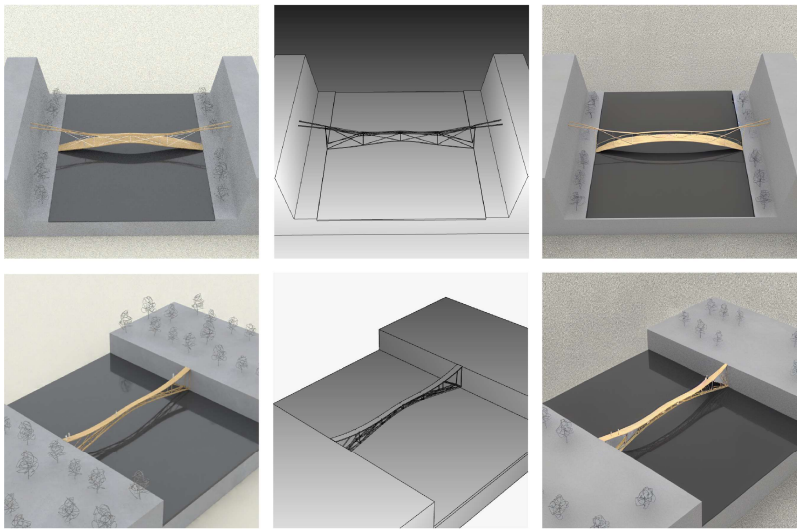



Figure 9: Guided AI image generation via SE Toolkit with the self-trained LoRA model



a) b) c)

Figure 10: Visual Comparison of V-Ray-rendered images (column *a*) and self-trained LoRA-generated images (column *c*) with depth and canny ControlNet units used for the image generation (column *b*)

## 5. Conclusion

This case study illustrates the proposed DL-enabled Workflow and Structural Embodiment Toolkit's capabilities, notably through integrating the CEM Adaptor, exemplifying the connections with various form-finding tools. The toolkit then automates solid geometry generation for structures and site models.

Despite some quality trade-offs (Figure 10), AI visualisations via the toolkit offer considerable speed benefits, which are crucial for the conceptual design phase's iterative nature.

Moreover, the toolkit not only refines existing workflows but also fosters a self-reinforcing ecosystem by easing the task of creating AI model training datasets. The publicly available LoRA model, SE_CEM_BRIDGE, underscores the project's success and deep learning's transformative impact on structural design [42].

However, the toolkit is not without its limitations. The quality of AI-generated visualisations and the current dependency on specific form-finding tools pose challenges that could be addressed in future versions. Looking forward, expanding the range of form-finding adaptors and encouraging community-driven development are crucial steps towards enhancing the toolkit's versatility and industry relevance. Additionally, obtaining more comprehensive feedback from industry peers will be essential to ensure the toolkit's evolution reflects a broader spectrum of user needs and preferences, further bridging the gap between computational design and practical application.

## References

[1] C. Mishra and D. L. Gupta, "Deep Machine Learning and Neural Networks: An Overview," in *Int. J. Hybrid Inf. Technol.*, vol. 9, no. 11, pp. 401–414, 2016.
[2] A. Jamwal, R. Agrawal, and M. Sharma, "Deep learning for manufacturing sustainability: Models, applications in Industry 4.0 and implications," in *Int. J. Inf. Manag. Data Insights*, vol. 2, no. 2, p. 100107, Nov. 2022.
[3] J. Bielski, B. Mete, C. Langenhan, F. Petzold, V. Eisenstadt, and K.-D. Althoff, "CHASING THE WHITE RABBIT - A case study of predicting design phases of architects by training a deep neural network with sketch recognition through a digital drawing board," in *International Conferences on Computational Creativity*, Jun. 2022.
[4] Z. Zheng, J. Li, L. Zhu, H. Li, F. Petzold, and P. Tan, "GAT-CADNet: Graph Attention Network for Panoptic Symbol Spotting in CAD Drawings." *ArXiv*, Jan. 10, 2022.
[5] G. Boller and P. D'Acunto, "Structural design via form finding: Comparing Frei Otto, Heinz Isler and Sergio Musmeci," in *History of Construction Cultures Volume 2*, CRC Press, 2021.
[6] D. Piker, "Kangaroo Physics." Accessed: Feb. 15, 2024. [Online]. Available: https://www.food4rhino.com/en/app/kangaroo-physics
[7] M. Rippmann, "Funicular Shell Design: Geometric approaches to form finding and fabrication of discrete funicular structures," ETH Zurich, 2016.
[8] T. Oberbichler, "oberbichler/EQlib." Aug. 06, 2023. Accessed: Feb. 15, 2024. [Online]. Available: https://github.com/oberbichler/EQlib
[9] one!3d, "Kiwi!3D." Accessed: Feb. 15, 2024. [Online]. Available: https://www.kiwi3d.com/theory/
[10] "SOFiSTiK | 2024 Highlights: New Features for Efficient BIM and Structural Analysis." Accessed: Feb. 15, 2024. [Online]. Available: https://www.sofistik.com/highlights-sofistik-2024
[11] "Software for Form Finding, Statics and Cutting Pattern Generation of membrane- and cable net structures." Accessed: Feb. 15, 2024. [Online]. Available: https://www.technet-gmbh.com/en/products/easy/
[12] Mediaworks, "GSA: Structural Analysis and Design Software," Oasys. Accessed: Feb. 15, 2024. [Online]. Available: https://www.oasys-software.com/products/gsa/
[13] P. D'Acunto, J. P. Jasienski, P. O. Ohlbrock, C. Fivet, J. Schwartz, and D. Zastavni, "Vector-based 3D graphic statics: A framework for the design of spatial structures based on the relation between form and forces," in *Int. J. Solids Struct.*, vol. 167, pp. 58–70, Aug. 2019.
[14] P. O. Ohlbrock and P. D'Acunto, "A Computer-Aided Approach to Equilibrium Design Based on Graphic Statics and Combinatorial Variations," in *Comput.-Aided Des.*, vol. 121, 2020.
[15] S. Yuchi, P. D'Acunto, J.-P. Jasienski, and P. O. Ohlbrock, "A new tool for the conceptual design of structures in equilibrium based on graphic statics," in *fib Symposium Proceedings* Sep. 2021, pp. 445–446.
[16] L. Fuhrimann, V. Moosavi, P. O. Ohlbrock, and P. Dacunto, "Data-Driven Design: Exploring new Structural Forms using Machine Learning and Graphic Statics." *ArXiv*, Sep. 23, 2018.

[17] J. de Miguel, M. E. Villafane, L. Piskorec, and F. Sancho-Caparrini, "Deep Form Finding Using Variational Autoencoders for deep form finding of structural typologies," in *ECAADE SIGRADI 2019, VOL 1*, eCAADe, 2019, pp. 71–80.
[18] F. Bertagna, P. D'Acunto, and P. O. Ohlbrock, "Conceptual design of three-dimensional structural and sun-shading façades supported by machine-learning," in *Proc. IASS Annu. Symp.*, vol. 2020, no. 8, pp. 1–11, 2020.
[19] K. S. Ochoa, P. O. Ohlbrock, P. D'Acunto, and V. Moosavi, "Beyond typologies, beyond optimization: Exploring novel structural forms at the interface of human and machine intelligence," *Int. J. Archit. Comput.*, vol. 19, no. 3, pp. 466–490, Sep. 2021.
[20] Z. Guo, K. Saldana Ochoa, and P. D'Acunto, "Enhancing structural form-finding through a text-based AI engine coupled with computational graphic statics," in *Proceedings of IASS Annual Symposia*, Sep. 2022.
[21] L. Bleker, R. Pastrana, P. O. Ohlbrock, and P. D'Acunto, "Structural Form-Finding Enhanced by Graph Neural Networks," in *Towards Radical Regeneration*, C. Gengnagel, O. Baverel, G. Betti, M. Popescu, M. R. Thomsen, and J. Wurm, Eds., Cham: Springer International Publishing, 2023.
[22] OpenAI, "ChatGPT." Accessed: Feb. 21, 2024. [Online]. Available: https://chat.openai.com
[23] B. C. Yin, W. T. Wang, and L. C. Wang, "Review of Deep Learning," in *Beijing Gongye Daxue XuebaoJournal Beijing Univ. Technol.*, vol. 41, no. 1, pp. 48–59, Apr. 2018.
[24] OpenAI, "DALL·E 3." Accessed: Feb. 21, 2024. [Online]. Available: https://openai.com/dall-e-3
[25] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, "Hierarchical Text-Conditional Image Generation with CLIP Latents." *ArXiv*, Apr. 12, 2022.
[26] Midjourney. Accessed: Feb. 21, 2024. [Online]. Available: https://www.midjourney.com/home
[27] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-Resolution Image Synthesis with Latent Diffusion Models," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2022-June, pp. 10674–10685, 2022.
[28] L. Zhang and M. Agrawala, "Adding Conditional Control to Text-to-Image Diffusion Models," in *ArXiv Pre-Print Serv.*, 2023.
[29] X. Pan, A. Tewari, T. Leimkühler, L. Liu, A. Meka, and C. Theobalt, "Drag Your GAN: Interactive Point-based Manipulation on the Generative Image Manifold," *ArXiv*, 2023.
[30] ArkoAI. Accessed: Feb. 21, 2024. [Online]. Available: https://arko.ai/
[31] EvolveLAB, "VERAS". Accessed: Feb. 21, 2024. [Online]. Available: https://www.evolvelab.io/veras
[32] E. J. Hu *et al.*, "LoRA: Low-Rank Adaptation of Large Language Models." *ArXiv*, Oct. 16, 2021.
[33] S. Luo *et al.*, "LCM-LoRA: A Universal Stable-Diffusion Acceleration Module." *ArXiv*, 2023.
[34] comfyanonymous, "comfyanonymous/ComfyUI." Feb. 14, 2024. Accessed: Feb. 14, 2024. [Online]. Available: https://github.com/comfyanonymous/ComfyUI
[35] AUTOMATIC1111, "Stable Diffusion Web UI." Aug. 2022. Accessed: Feb. 14, 2024. [Online]. Available: https://github.com/AUTOMATIC1111/stable-diffusion-webui
[36] ambrosinus, "Ambrosinus Toolkit." Accessed: Feb. 14, 2024. [Online]. Available: https://ambrosinus.altervista.org/blog/ambrosinus-toolkit/
[37] "RhinoCommon 8.0.23304.9001." Accessed: Apr. 14, 2024. [Online]. Available: https://nuget.org/packages/RhinoCommon/
[38] T. Sun, "LupoSun/StructuralEmbodimentToolkit." Mar. 04, 2024. Accessed: Mar. 16, 2024. [Online]. Available: https://github.com/LupoSun/StructuralEmbodimentToolkit
[39] "3D Rendering Software - V-Ray | Chaos." Accessed: Apr. 14, 2024. [Online]. Available: https://www.chaos.com/3d-rendering-software
[40] bmaltais, "bmaltais/kohya_ss." Accessed: Mar. 16, 2024. [Online]. Available: https://github.com/bmaltais/kohya_ss
[41] Runway, "runwayml/stable-diffusion." Runway, Feb. 17, 2024. Accessed: Feb. 17, 2024. [Online]. Available: https://github.com/runwayml/stable-diffusion
[42] T. Sun, "LupoSun/SE_CEM_BRIDGE · Hugging Face." Accessed: Mar. 16, 2024. [Online]. Available: https://huggingface.co/LupoSun/SE_CEM_BRIDGE