Proceedings of the IASS Annual Symposium 2025
"The living past as a source of innovation"
27 October– 31 October 2025, México City, México>
<Juan Gerardo Oliva, Juan Ignacio del Cueto, Elisa Drago (eds.)

Automated Visualization for Structural Form-Finding using Orchestrated Multimodal Machine Learning Agents

Tao Sun*, Lazlo Bleker[a], Mustafa Cem Günes[a], Pierluigi D'Acunto[a]

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

[a] Professorship of Structural Design, Technical University of Munich

Abstract

Effective collaboration is fundamental to successful structural design. While engineers rely on numerical models and technical diagrams, architects prefer conceptual visualizations conveying design intent. Traditional rendering is laborious, hindering the rapid iteration in conceptual design. Although recent text-to-image machine learning (ML) models offer a promising alternative, existing workflows often require manual intervention to correct unnatural artifacts. This paper addresses this gap by proposing a novel, agent-workflow hybrid framework that introduces autonomous self-correction to the visualization process. We introduce an architecture where multiple ML agents are orchestrated in a reflective loop to autonomously generate, evaluate, and iteratively refine visualizations. Using a CAD view and text description as input, these multimodal agents collaborate to generate images. If the result is flawed, the system autonomously uses the critique to guide a new generation attempt. This agentic approach reduces manual rework and consistently produces high-fidelity results, enabling a more efficient and creative design exploration.

Keywords: Conceptual Structural Design, Architectural Visualization, Form-finding, Generative AI, Large Language Model, Multi-Agent System

1. Introduction

1.1 AI-Informed Structural Design

The application of artificial intelligence (AI) in structural engineering has seen significant growth in recent years, with most contributions focusing on structural analysis and optimization. Researchers have successfully employed machine learning models as surrogates for computationally expensive finite element analyses or to accelerate complex design tasks[1]. For instance, in structural analysis, Nie *et al.* demonstrated the use of convolutional neural networks for predicting stress fields in cantilevered structures [2], offering a deep learning-based alternative to classical methods. In the domain of optimization, Yu *et al.* proposed a novel deep learning method to determine near-optimal topological designs without iterative schemes [3].

In parallel, a growing body of research is exploring the use of AI for the more intuitive and open-ended challenges of conceptual design and form-finding. This emerging sub-field aims to leverage AI to explore novel structural forms and enhance design creativity. Early explorations in this direction by Danhaive and Mueller introduced methods for design subspace learning, using performance-conditioned generative modeling to systematically explore diverse structural design spaces [4]; Saldana Ochoa *et al.* focused on the interface of human and machine intelligence to discover forms beyond conventional typologies and optimization [5], while Tam *et al.* focused on geometric deep learning for trans-

topological design exploration of reticulated equilibrium shell structures [6]. Later, Bleker *et al.* integrated graph neural networks (GNNs) into form-finding workflows to automate tasks like labeling and interpreting topology graphs in equilibrium models [7], following another contribution to Logic-Informed Graph Neural Networks (LIGNN) that integrates the validity conditions of Combinatorial Equilibrium Modelling (CEM) [8] into the learning process [9]. More recently they proposed an E(3)-Equivariant Graph Neural Network (EGNN) architecture for inverse form-finding [10]. Further exploring the acceleration of design processes, Pastrana *et al.* combined neural networks with differentiable mechanics simulators to enable real-time design of architectural structures while explicitly guaranteeing mechanical integrity [11]. Collectively, these contributions signify a rapid development towards leveraging AI for more efficient and innovative structural solutions. However, the structural design process encompasses more than form-finding and analysis, presenting further opportunities for AI integration.

1.2 AI for Image Generation

After structural form-finding, the conceptual structural design process also involves visualization as the subsequent step [12]. While form-finding determines the optimal shape and solid geometry generation defines the tangible members, visualization is crucial for communicating the design intent. In the conceptual phase, characterized by rapid iterations and frequent exchange between diverse stakeholders, the ability to quickly generate compelling and easily understandable visualizations from precise engineering calculations is essential. This paper focuses on enhancing this third step.

The landscape of AI-driven image generation has evolved dramatically since the advent of models like Stable Diffusion 1.5 [13] in 2022. Recent advancements offer significant improvements in image quality, user control, and generation speed. The release of SDXL [14] in July 2023 offered native 1024x1024 resolution and enhanced photorealism compared to its predecessors. Stable Diffusion (SD) 3 [15] and SD 3.5 [16], further pushed boundaries with their transformer-based architecture, improving multi-subject prompting and typography. The FLUX.1 series [17] from Black Forest Labs set new benchmarks in image quality, prompt adherence, and typography. From March 2025, OpenAI's GPT-4o [18] introduced native image generation capabilities, enabling higher visual generation quality and nuanced understanding of multimodal prompts.

Alongside quality, control mechanisms have become more sophisticated. ControlNet [19], introduced for Stable Diffusion in 2023, allows for precise conditional guidance using inputs like depth maps, canny edges, or semantic segmentation, ensuring greater fidelity when visualizing structural forms. In 2024, the FLUX.1 series offered a group of specialized models, including Depth and Canny, providing control directly from its flow matching transformer architecture [20] - a different mechanism than ControlNet. On the commercial, closed-source side, GPT-4o's conversational image modification capabilities allow users to refine images iteratively through natural language dialogue, offering intuitive and precise adjustments. These advancements are crucial for generating visualizations that accurately reflect engineering designs while meeting aesthetic requirements.

Generation speed has also seen remarkable improvements. Techniques like Latent Consistency Models (LCMs) significantly accelerated earlier diffusion models [21]. More recently, techniques like 4-bit quantization (SVDQuant) [22], have further reduced latency and memory usage for advanced models like FLUX, enabling near real-time generation.

1.3 Multimodal Agents and Reasoning

The advancements in AI capabilities have paved the way for a new paradigm: agentic AI. But the concept of agentic AI is not without historical precedent; it is deeply rooted in decades of AI research on "intelligent agents". In 1995, Wooldridge and Jennings provided a classic definition of an intelligent agent as "a computing system situated in an environment, capable of flexible and autonomous action to meet its design objectives" [23]; in the same year, Russell and Norvig defined an agent as "anything that

can be viewed as perceiving its environment through sensors and acting upon that environment through actuators" [24]. Echoing to these ideas, emerging agentic AI systems are designed to make autonomous decisions, take actions in their environment, and learn from their experiences, moving beyond simple pattern recognition to more complex problem-solving [25]. A core mechanism enabling this is function calling, where Large Language Models (LLMs) can intelligently determine when to call external tools or APIs and formulate the necessary parameters based on a user's request. To standardize these interactions, the Model Context Protocol (MCP) was introduced as a universal interface between AI models and external tools by Anthropic in late 2024 [26]. Central to these agents are the core capabilities of multimodality, the ability to process and correlate information from different data types like text and images [27], and reasoning, the capacity to break down problems into sequential steps to derive solutions, often elicited through techniques like chain-of-thought prompting [28]. By having these abilities, multiple AI agents, each with specialized skills or tools, can work together to tackle complex tasks that would be beyond the capabilities of a single agent. Frameworks like AutoGen from Microsoft Research exemplify this by enabling the creation of applications where multiple agents can communicate and coordinate their actions to achieve a common goal [29].

1.4 Previous Work and Motivation

Our previous work, "Structural Embodiment" [12], introduced a toolkit that integrated form-finding with AI-powered visualization on the McNeel Grasshopper platform [30]. However, this toolkit shared a critical limitation with the current state-of-the-art in commercial AI visualization tools like LookX.ai [31] and EvolveLAB's VERAS [32]: it was fundamentally a linear, one-shot execution. These systems function as user-driven instruments, executing a prompt once and relying entirely on a human-in-the-loop for quality control, requiring to manually adjust settings and re-initiate the generation process.

This paper is directly motivated by this research gap. We move beyond the one-shot paradigm by introducing a novel, extendable multimodal agentic framework. The core contribution is a system capable of reflective self-correction, where ML agents can autonomously evaluate a generated visualization and, if it is flawed, re-strategize to improve the outcome without user intervention. By leveraging a more flexible backend**,** faster generation models, and an orchestration of specialized agents, our framework aims to automate the creation of high-quality, context-aware visualizations, fundamentally reducing the need for manual oversight and enhancing the creative efficiency of the conceptual design process.

2. Agent-Workflow Hybrid Framework

There are multiple ways of interacting with a multi-agent system. For instance, one of the common arrangements is designed for conversational applications like chatbots, where the focus is on interactive, turn-by-turn dialogue with a user. A workflow, on the other hand, is built for process automation with less user intervention. By combining AI agents and a structured workflow, the system can strike a balance between the creativity of the agents and the stability of processes [33].

Shown in Figure 1, we propose an agent-workflow hybrid framework that converts the initial 3D view from the form-finding pipeline into a final visualization. The Dreamer ingests the view with a text prompt and produces depth conditioning while "re-imagining" the scene; the Generator invokes the appropriate tools to generate images; the Evaluator critiques each result and flags artefacts; and the Memory stores validated variables and feedback to guide iterative refinement. A detailed demonstration is provided in Section 3.2.

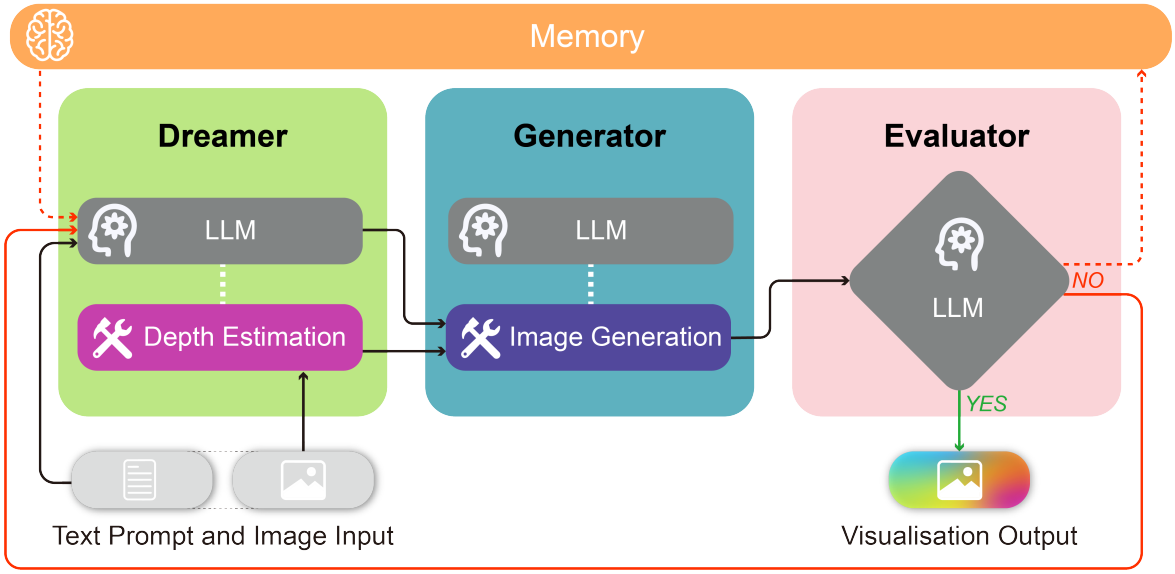

Figure 1: Orchestration diagram of the proposed agent-workflow hybrid framework, illustrating the interaction between the Dreamer, Generator, and Evaluator agents.

2.1. System Specification

Tables 1–5 adopt the Figure 1 color scheme: each color denotes the same workflow element. For example, orange variables in Table 1 are values stored in Memory; the green in Table 2 corresponds to the Dreamer agent; magenta highlights the local sub-workflow a (depth estimation).

Name	Type	Explanation
cad view	File (png)	The view of the form-found structure from CAD program
depth map	File (png)	(Memory) Input / generated depth map of *cad view*
visualization	File (png)	The generated image
description	String	A description for the structure and the scene
dream	String	The AI-enriched text prompt for image generation
visualizations	Array of Strings	(Memory) A list storing the generated images encoded as Base64
bad dreams	Array of Strings	(Memory) A list of text prompts that resulted in unsatisfactory images
eval results	Array of Strings	(Memory) A list recording the evaluation results for generated images

Table 1: The list of system variables in the workflow.

The framework's agents are assigned distinct roles, with their underlying LLMs carefully selected based on the capabilities required for each task. This approach optimizes for both cost and performance, as using a model with unnecessary capabilities—for instance, a vision-enabled model for a task that only requires function calling—would be inefficient. To clearly define each agent's role and interactions, we use the standardized specification format detailed in (Table 2, 3 and 4). For brevity, the full system prompts for each agent are omitted here but are available in the project's public repository [34].

2.1.1. Dreamer

Name	Dreamer
Role	Prepare text prompt and image conditioning
LLM Ability	Vision | Reasoning | Function Calling
Tools	Local sub-workflow a
Input	*cad view* | *description* | *bad dreams*
Task	- Describe the structure in detail based on *cad view* and *description* - Create a detailed *dream* by reimagining the structure in the desired scene from *description* - Optimize the *dream* according to *bad dreams* - **If not present:** create *depth map* by using the tool
Output	*dream* | *depth map*

Table 2: Personal information for the agent "Dreamer"

2.1.2. Generator

Name	Generator
Role	Generate appealing and accurate image of the given structure
LLM Ability	Function Calling
Tools	Local sub-workflow b
Input	*dream* | *depth map* | *visualizations*
Task	- Generate a *visualization* by using the tool based on *dream* and *depth map* - Put the *visualization* on top of the *visualizations*
Output	*visualizations*

Table 3: Personal information for the agent "Generator"

2.1.3. Evaluator

Name	Evaluator
Role	Evaluate generation outcome, record experience, coordinate the next step
LLM Ability	Vision | Reasoning
Tools	-
Input	*visualizations* | *description* | *bad dreams* | *eval results*
Task	- Evaluate if the first image in *visualizations* fulfills the *description* - Evaluate if the first image in *visualizations* has no unnatural AI artifacts - Record the evaluation result on top of *eval results* **If both evaluations are positive:** Output the final image **Else:** Record the dream on top of *bad dreams*, go back to Dreamer to iterate
Output	*visualizations* | *bad dreams* | *eval results*

Table 4: Personal information for the agent "Evaluator"

2.2. Workflow

2.2.1 Global Workflow for the agents

The global workflow orchestrates the three agents in a structured, reflective loop that begins with user input and culminates in a satisfactory visualization. The process is initiated when the Dreamer agent receives the initial *cad view* and *description*. It first interprets these inputs to generate a detailed prompt, or *dream*, and uses its local tool to produce a corresponding *depth map* for geometric control. These are then passed to the Generator, which executes its own local non-agentic workflow via function calling to synthesize a candidate image. This image is immediately forwarded to the Evaluator, the core decision-making component of the system. The Evaluator assesses the image against the initial *description* and

checks for any unnatural visual artifacts. If the image is deemed successful, the workflow concludes and outputs the final image. However, if it fails, the Evaluator initiates a feedback loop: it records the flawed *dream* into a persistent *bad dreams* list and reinvokes the Dreamer. Armed with this new contextual knowledge of what to avoid, the Dreamer generates a revised *dream*, restarting the cycle. This iterative process of generation, evaluation, and guided refinement allows the system to autonomously learn from its mistakes and converge towards a high-quality visualization that aligns with the user's original intent.

2.2.2 Local Sub-Workflow as Tools

A key feature of our framework is the use of local sub-workflows to handle specialized, computationally intensive tasks (Table 5). These are predefined, non-agentic pipelines that run independently of the agents' reasoning processes. By encapsulating complex operations—such as depth estimation or image synthesis—into discrete tools, the agents can orchestrate them through simple function calls. This abstraction allows the global workflow to remain focused on high-level strategy and decision-making, while leveraging the efficiency of optimized, task-specific processes.

Sub-Workflow	a	b
Purpose	Depth Estimation	Image Synthesis based on text prompt guided by depth information
Tool Owner	Dreamer	Generator
Input	*cad view*	*dream* | *depth map*
Output	*depth map*	*visualization*

Table 5: Specifications of the non-agentic local sub-workflow

3. Case Study

To demonstrate our framework, we perform a case study using a synthetic dataset of 50-meter-long bridges generated via form-finding based on CEM. The dataset contains numerous bridges, each with a distinct design configuration, allowing us to showcase the system's visualization capabilities on complex, form-found structures.

3.1 System Stack

We prototyped the system on the open-source node-based agentic AI development platform Dify v1.4.1 [35] for the high-level agentic workflow. For function calling, we deployed ComfyUI v0.3.40 [36] to establish local sub-workflows as tools available for the agents.

Dify, ComfyUI and all the local ML models are hosted on a consumer level Windows workstation equipped with an Intel Core i7-11700K CPU and an NVIDIA GeForce RTX 5080 GPU. The other cloud-based models are accessed via APIs of their respective providers. The detailed LLMs and ML models involved are stated in Table 6.

Agent / Tool	**Model**	**Access Method**
Dreamer	Gemini 2.5 Flash Preview 05-20 [37]	Gemini API
Generator	DeepSeek-V3-0324 [38]	deepseek Platform API
Evaluator	gpt-4o [39]	Azure OpenAI API (2024-12-01-preview)
Sub-workflow a	depth_anything_v2_vitl_fp32 [40]	Local
Sub-workflow b	svdq-fp4-flux.1-depth-dev [41]	Local

Table 6: ML models involved in the case study

The prototype is publicly available on a GitHub repository [34], All the involved models are swappable if their technical requirements meet the framework's specifications stated in Section 2.1. This prototype also supports API access, enabling integration into structural design processes.

3.2 Iterative Self-Correcting Generation

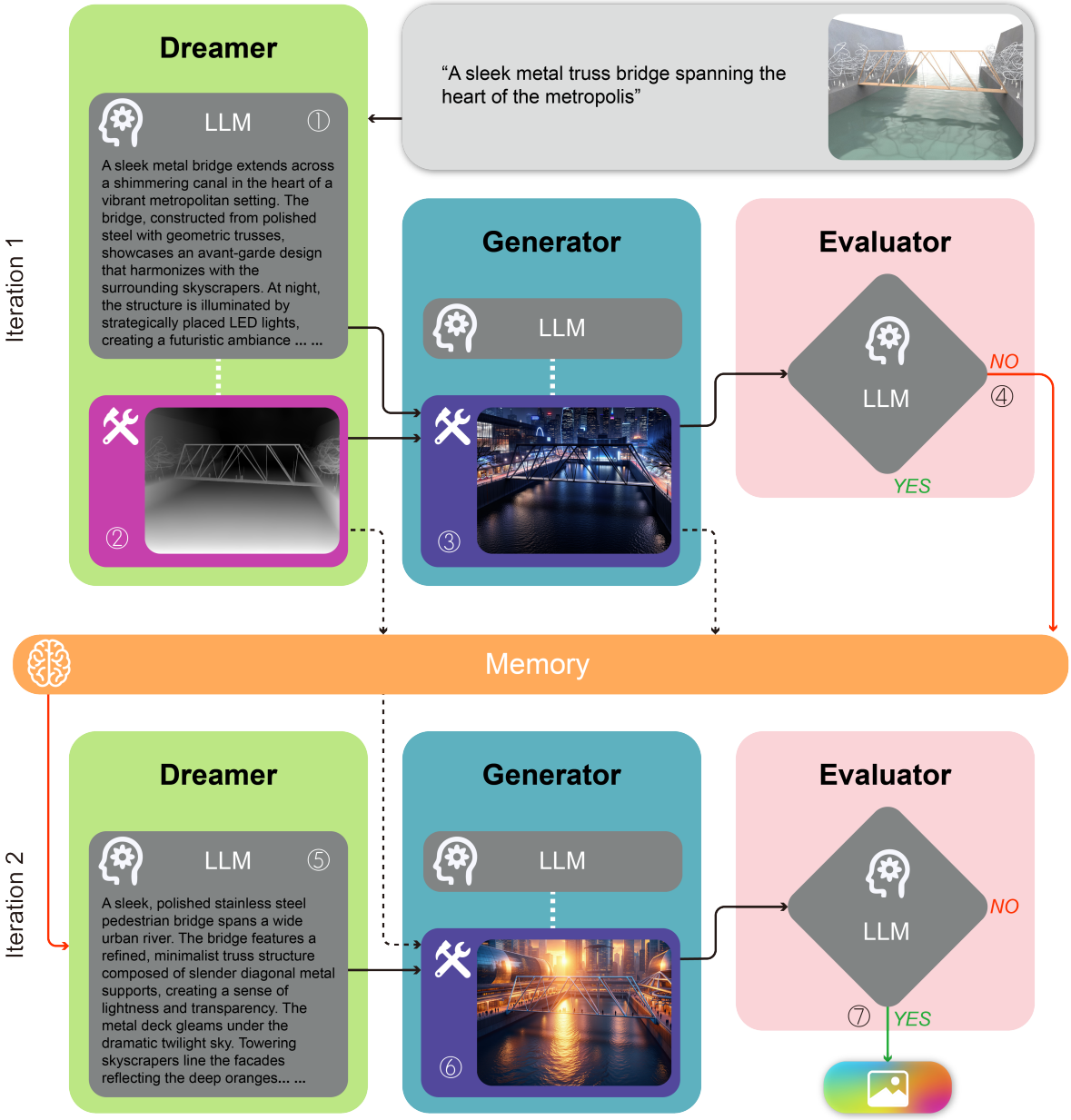

Figure 2: Demonstration of the framework's autonomous, self-correction process. Iteration 1 shows an initial failure due to visual artifacts, which is identified by the Evaluator. The system then uses this failure, stored in Memory, to inform a successful second iteration, highlighting the framework's ability to learn from its mistakes without user intervention.

To demonstrate the framework's core capability of self-correction, we now document the generation process for a truss bridge structure, as illustrated in Figure 2. The system was provided with a *cad view* and the high-level *description*: "A sleek metal truss bridge spanning the heart of the metropolis."

Iteration 1: Initial Generation

In the initial cycle, the Dreamer agent processed the inputs. It first performed a visual analysis of the *cad view* to formulate a descriptive dream (① 7.4s) based on *description* and then called its local tool to generate a corresponding *depth map* (② 8.2s). Subsequently, the Generator agent executed its own tool, using the *dream* and *depth map* to synthesize a candidate *visualization* (③ 20.8s). However, upon inspection, the Evaluator agent identified significant visual artifacts—specifically, "broken edges" on the structural members—and returned a "NO" verdict (④ 4.7s). This failure triggered the feedback loop, adding the flawed prompt to the *bad dreams* in Memory along with other recorded variables like *eval results, visualizations*. The total time for this unsuccessful first iteration was 41.1 seconds.

Iteration 2: Refinement and Successful Generation

The workflow was then automatically re-initiated. Guided by the *bad dreams* in Memory, the Dreamer agent formulated a revised *dream* aimed at correcting the previous artifacts (⑤ 7.3s). Crucially, the system bypassed *depth map* re-generation, as it was retained from the first cycle. This refined guidance was passed to the Generator, which produced a new *visualization* (⑥ 25.1s), reusing the same random seed as the first attempt to ensure a more controlled, gradual refinement. This second iteration was successful; the resulting image was free of obvious artifacts and featured a clear structure with realistic materials. The Evaluator confirmed this positive outcome with a "YES" verdict (⑦ 5.2s), successfully concluding the workflow in a total time of 37.6 seconds.

Comparison to Our Old Method

A direct comparison to our previous one-shot generation method highlights the significant advantages of the proposed agentic framework. The older method required considerable manual intervention; on average, it took six manual regeneration attempts (54.3s in total) to obtain a single acceptable image. Even then, the result contains visual artifacts, such as inconsistent member thickness, distorted or "wobbly" geometry, and unrealistic background elements. In contrast, while the new autonomous framework took longer for the successful two-iteration process (76.8s in total), it required user input only once at the beginning. As shown in Figure 3, the final output is of higher quality, demonstrating superior geometric fidelity and realism.

Furthermore, to demonstrate its broader efficacy, the framework was successfully applied to other form-found structures of varying typologies (Figure 4), consistently yielding high-quality visualizations.

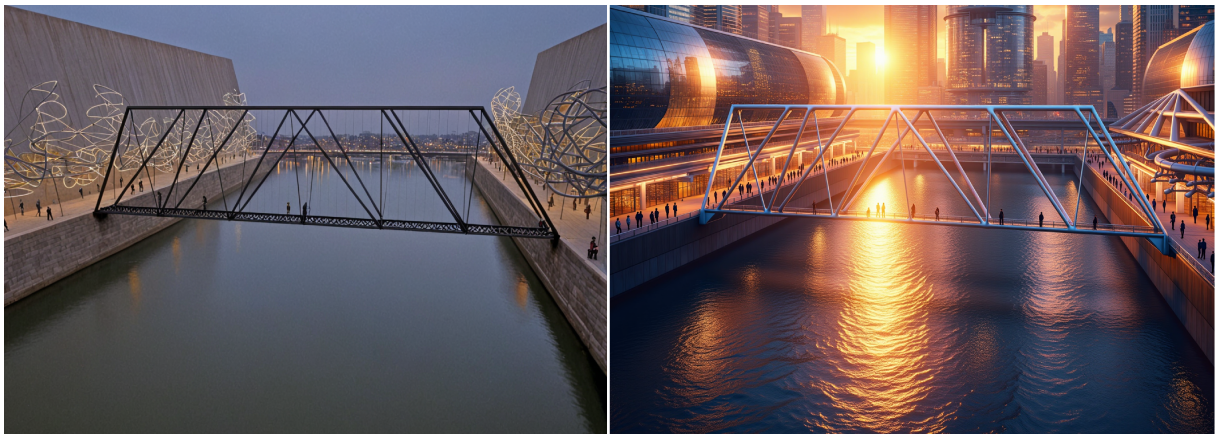

Figure 3: A comparison of the final visualization output, demonstrating the improved quality of the proposed agentic framework (right) over our previous one-shot method (left)

4. Conclusion

In this paper, we presented an open-source, agent-workflow hybrid framework capable of autonomous self-correction. We have shown that by orchestrating a team of specialized ML agents in a reflective loop, a system can move beyond the rigid, one-shot paradigm of previous methods. The framework autonomously generates, evaluates, and iteratively refines images, learning from its own failed attempts to achieve a desired outcome. As demonstrated in our case study, this agentic approach yields substantial improvements in visual quality and geometric fidelity while reducing the unnatural artifacts common in prior approaches. The framework thereby enhances communication between stakeholders and empowers designers to focus on innovation during the crucial conceptual stages of structural design.

Despite these promising results, we acknowledge the framework's current limitations. The Evaluator agent's quality control, while effective, does not yet guarantee perfect structural accuracy due to inherent constraints in current generative models. Furthermore, the agents' autonomy is currently limited to a predefined set of tools. Future work should therefore focus on two key areas: improving the precision of the evaluation mechanism and expanding agent autonomy by providing them with a broader and more dynamic choice of tools and workflows.

Figure 4: Visualization examples for various form-found structures from the synthetic dataset, generated using the proposed agentic framework with system specifications as detailed in Section 3.1.

References

[1] H. T. Thai, “Machine learning for structural engineering: A state-of-the-art review,” *Structures*, vol. 38, pp. 448–491, Apr. 2022, doi: 10.1016/J.ISTRUC.2022.02.003.

[2] Z. Nie, H. Jiang, and L. B. Kara, “Stress Field Prediction in Cantilevered Structures Using Convolutional Neural Networks,” *J Comput Inf Sci Eng*, vol. 20, no. 1, Aug. 2018, doi: 10.1115/1.4044097.

[3] Y. Yu, T. Hur, J. Jung, and I. G. Jang, “Deep learning for determining a near-optimal topological design without any iteration,” *Structural and Multidisciplinary Optimization*, vol. 59, no. 3, pp. 787–799, Mar. 2019, doi: 10.1007/S00158-018-2101-5/METRICS.

[4] R. Danhaive and C. T. Mueller, "Design subspace learning: Structural design space exploration using performance-conditioned generative modeling," *Autom Constr*, vol. 127, p. 103664, Jul. 2021, doi: 10.1016/J.AUTCON.2021.103664.

[5] K. Saldana 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," *International Journal of Architectural Computing*, vol. 19, no. 3, pp. 466–490, Sep. 2021, doi: 10.1177/1478077120943062.

[6] K.-M. M. Tam, V. Moosavi, T. Van Mele, and P. Block, "Towards trans-topological design exploration of reticulated equilibrium shell structures with graph convolution networks," in *Proceedings of the IASS Annual Symposium 2020/21 and the 7th International Conference on Spatial Structures*, International Association for Shell and Spatial Structures (IASS), Aug. 2021.

[7] L. Bleker, R. Pastrana, P. O. Ohlbrock, and P. D'Acunto, "Structural Form-Finding Enhanced by Graph Neural Networks," *Towards Radical Regeneration: Design Modelling Symposium Berlin 2022*, pp. 24–35, Sep. 2023, doi: 10.1007/978-3-031-13249-0_3.

[8] P. O. Ohlbrock and P. D'Acunto, "A Computer-Aided Approach to Equilibrium Design Based on Graphic Statics and Combinatorial Variations," *Computer-Aided Design*, vol. 121, p. 102802, Apr. 2020, doi: 10.1016/J.CAD.2019.102802.

[9] L. Bleker, K. M. M. Tam, and P. D'Acunto, "Logic-Informed Graph Neural Networks for Structural Form-Finding," *Advanced Engineering Informatics*, vol. 61, p. 102510, Aug. 2024, doi: 10.1016/J.AEI.2024.102510.

[10] L. Bleker, P. D'Acunto, and K.-M. M. Tam, "Generalized Inverse Structural Form-Finding Using E(3)-Equivariant Physics-Based Graph Neural Networks," 2025, doi: 10.2139/SSRN.5208576.

[11] R. Pastrana, E. Medina, I. M. De Oliveira, S. Adriaenssens, and R. P. Adams, "Real-time Design of Architectural Structures with Differentiable Mechanics and Neural Networks," in *International Conference on Learning Representations (ICLR)*, 2025.

[12] T. Sun, P. D'Acunto, and F. Petzold, "Structural Embodiment-Unified Workflow and Toolkit for Form-finding, Solid Geometry Generation and Visualisation via Deep Learning Methods," in *Proceedings of the IASS 2024 Symposium*, P. Block, G. Boller, C. DeWolf, J. Pauli, and W. Kaufmann, Eds., Zurich, Aug. 2024.

[13] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-Resolution Image Synthesis with Latent Diffusion Models," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2022-June, pp. 10674–10685, Dec. 2021, doi: 10.1109/CVPR52688.2022.01042.

[14] D. Podell *et al.*, "SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis," *12th International Conference on Learning Representations, ICLR 2024*, Jul. 2023.

[15] P. Esser *et al.*, "Scaling Rectified Flow Transformers for High-Resolution Image Synthesis," *Proc Mach Learn Res*, vol. 235, pp. 12606–12633, Mar. 2024.

[16] stability.ai, "Stable Diffusion 3.5," Oct. 22, 2024. Accessed: Jun. 06, 2025. [Online]. Available: https://stability.ai/news/introducing-stable-diffusion-3-5

[17] Black Forest Labs, "FLUX," 2024. Accessed: Jun. 06, 2025. [Online]. Available: https://github.com/black-forest-labs/flux

[18] OpenAI, "GPT-4o System Card," Oct. 2024, Accessed: Jun. 06, 2025. [Online]. Available: https://arxiv.org/pdf/2410.21276

[19] L. Zhang, A. Rao, and M. Agrawala, "Adding Conditional Control to Text-to-Image Diffusion Models," *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3813–3824, Feb. 2023, doi: 10.1109/ICCV51070.2023.00355.

[20] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow Matching for Generative Modeling," *11th International Conference on Learning Representations, ICLR 2023*, Oct. 2022.

[21] S. Luo, Y. Tan, L. Huang, J. Li, and H. Zhao, "Latent Consistency Models: Synthesizing High-Resolution Images with Few-Step Inference," Oct. 2023, Accessed: Jun. 06, 2025. [Online]. Available: https://arxiv.org/pdf/2310.04378

[22] M. Li *et al.*, "SVDQuant: Absorbing Outliers by Low-Rank Components for 4-Bit Diffusion Models," Nov. 2024, Accessed: Jun. 06, 2025. [Online]. Available: https://arxiv.org/pdf/2411.05007

[23] M. Wooldridge and N. R. Jennings, "Intelligent agents: theory and practice," *Knowl Eng Rev*, vol. 10, no. 2, pp. 115–152, 1995, doi: 10.1017/S0269888900008122.

[24] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, no. 2. Prentice Hall, 1995.

[25] D. B. Acharya, K. Kuppan, and B. Divya, "Agentic AI: Autonomous Intelligence for Complex Goals - A Comprehensive Survey," *IEEE Access*, 2025, doi: 10.1109/ACCESS.2025.3532853.

[26] Anthropic, "Introducing the Model Context Protocol." Accessed: Jun. 06, 2025. [Online]. Available: https://www.anthropic.com/news/model-context-protocol

[27] J. B. Alayrac *et al.*, "Flamingo: a Visual Language Model for Few-Shot Learning," *Adv Neural Inf Process Syst*, vol. 35, Apr. 2022.

[28] J. Wei *et al.*, "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models," *Adv Neural Inf Process Syst*, vol. 35, Jan. 2022.

[29] V. Dibia *et al.*, "AutoGen Studio: A No-Code Developer Tool for Building and Debugging Multi-Agent Systems," Aug. 2024, Accessed: Jun. 06, 2025. [Online]. Available: https://arxiv.org/pdf/2408.15247

[30] McNeel & Associates, "Grasshopper." Accessed: Jun. 06, 2025. [Online]. Available: https://developer.rhino3d.com/guides/grasshopper/

[31] "LookX AI Cloud." Accessed: Jul. 11, 2025. [Online]. Available: https://www.lookx.ai/

[32] "VERAS | EvolveLAB." Accessed: Jul. 11, 2025. [Online]. Available: https://www.evolvelab.io/veras

[33] T. Wang, "Agent and Workflow Hybrid Architecture." Accessed: Jun. 06, 2025. [Online]. Available: https://mtda.cloud/en/blog/agent-workflow-hybrid-architecture/

[34] T. Sun, "AgenticAIVIS," May 2025, 1.0. Accessed: Jun. 06, 2025. [Online]. Available: https://github.com/LupoSun/AgenticAIVIS

[35] langgenius, "Dify," May 27, 2025, 1.4.1. Accessed: Jun. 06, 2025. [Online]. Available: https://github.com/langgenius/dify

[36] comfyanonymous, "ComfyUI," Mar. 2023. Accessed: Jun. 06, 2025. [Online]. Available: https://github.com/comfyanonymous/ComfyUI

[37] Google, "Gemini Models," 2025, 2.5 Flash Preview 05-20. Accessed: Jun. 08, 2025. [Online]. Available: https://ai.google.dev/gemini-api/docs/models

[38] DeepSeek, "DeepSeek-V3-0324 Release," 2025, DeepSeek-V3-0324 Release. Accessed: Jun. 08, 2025. [Online]. Available: https://api-docs.deepseek.com/news/news250325

[39] OpenAI, "Model - OpenAI API," 2025, 2024-12-01-preview. Accessed: Jun. 08, 2025. [Online]. Available: https://platform.openai.com/docs/models/gpt-4o

[40] Kijai, "DepthAnythingV2," Jun. 14, 2024, v2. Accessed: Jun. 08, 2025. [Online]. Available: https://huggingface.co/Kijai/DepthAnythingV2-safetensors/tree/main

[41] MIT Han Lab, "svdq-fp4-flux.1-depth-dev," Mar. 16, 2025. Accessed: Jun. 08, 2025. [Online]. Available: https://huggingface.co/mit-han-lab/svdq-fp4-flux.1-depth-dev/tree/main