Hierarchical Multi-Level 3D Geometry Generation with Stress-Aware Learning

Anonymous authors
Paper under double-blind review

Abstract

Current approaches for Lego 3d structural assembly are usually learned to maximize intersection over union between generated output and target construction. We propose a new approach which is able to build stable structures based on physics-aware reward. Our method employs a two-level agent architecture in which a high-level proximal policy optimization based planner proposes a scheme, while a low-level wave function collapse agent handles precise brick placement with constraint satisfaction. Experimental results demonstrate that our hierarchical method consistently constructs buildings that satisfy stress constraints while reducing material usage. We also show that replacing the computationally expensive finite element method solver with a fast Fourier neural operator achieves comparable performance, confirming the approach's scalability for large-scale problems.

1 Introduction

The construction of 3D structures from discrete building elements represents a fundamental challenge in robotics, automated manufacturing, and architectural design. Traditional approaches to this problem rely heavily on topology optimization methods that require extensive computational resources and often produce solutions that are difficult to construct in practice. Recent progress, particularly with generative models and reinforcement learning, has enabled the creation of complex shapes and assemblies Chung et al. (2021); Kolodiazhnyi et al. (2025). However, a significant gap remains between generating objects that are visually plausible and those that are physically viable. Most current approaches to LEGO brick construction focus on replicating a target shape, commonly employing geometric metrics such as intersection over union (IoU) as the main optimization criterion Chung et al. (2021). Although effective within their intended scope, these methods do not ensure that the resulting structure can support its own weight or resist external forces. We focus on the complex and application-driven challenge: structurally-informed combinatorial construction. The objective moves past simply assembling bricks to match a shape, aiming instead to create a stable configuration that complies with physical principles, reduces material usage, and maintains structural integrity under load. This problem introduces a new layer of complexity, as the validity of an action depends not only on local geometric constraints (e.g., element non-intersection) but also on its global impact on the entire structure's stress distribution.

The key insight driving our work is that effective structural construction requires reasoning at multiple levels of abstraction: strategic planning for overall structural layout and detailed execution for precise element placement. This naturally suggests a hierarchical approach in which different agents operate on different scales and optimize for different objectives. Our high-level planner focuses on strategic decisions about load distribution and structural stability, while our low-level executor ensures that individual brick placements satisfy local constraints.

Major Contributions:

- Novel Problem Formulation: We introduce stress minimization as a primary objective for 3D construction, moving beyond shape reconstruction to consider structural integrity and resource efficiency.

- Hierarchical Agent Architecture: We present a two-level system that integrates Proximal Policy Optimization (PPO) based strategic planning with Wave Function Collapse (WFC) based constraint satisfaction, enabling both global optimization and local constraint enforcement.

- Physics-Integrated Rewards: Our approach incorporates realistic stress analysis using finite element method (FEM), providing physics-based feedback for structural optimization.

- Efficient Constraint Handling: Using WFC for low-level execution, we achieve efficient constraint satisfaction without the computational overhead of traditional planning approaches.

We validate our framework through a series of experiments that test its core components. Our experiments demonstrate that the hierarchical 3D planner achieves higher success rates compared to a simplified 2D version, confirming that full spatial reasoning is critical to achieving structural stability. We then show that the 3D planner's high performance is maintained when a fast neural operator replaces the traditional physics solver, establishing the viability of our approach for scalable, structurally-aware construction.

2 Related Work

Recent advances in reinforcement learning (RL) based construction have focused primarily on LEGO brick assembly tasks. Brick-by-Brick introduced a pioneering approach that uses graph neural networks and action validity prediction for sequential brick placement, optimizing for IoU with target shapes Chung et al. (2021). Their work demonstrated that incomplete target information (2D images) could be sufficient for constructing complex 3D objects through a comprehensive understanding of partial information and long-term planning, but was limited to shape reconstruction objectives. Budget-aware construction was explored in BrECS, which combines U-shaped convolutional networks with efficient constraint satisfaction for sequential brick assembly Ahn et al. (2024). Although this approach showed improvements in assembly speed and constraint handling, it remained focused on shape completion rather than structural optimization. More recently, multimodal approaches have emerged with CADrille, which combines point clouds, images, and text for computer-aided design reconstruction Kolodiazhnyi et al. (2025). However, these approaches primarily target design reconstruction rather than structural construction with physical constraints. A framework Kakooee & Dillenburger (2024) leverages PPO algorithms to optimize space layouts, showing how RL can accommodate offline tasks and seamlessly integrate with existing computer-aided design software.

In addition to learning-based construction methods, the WFC algorithm has emerged as a powerful tool for procedural content generation, particularly in scenarios that require the adherence to local constraint rules. Originally developed for tile-based image generation, WFC operates by maintaining a superposition of possible states for each cell in a grid and progressively collapsing these possibilities based on local compatibility constraints Gumin (2016). The effectiveness of the algorithm arises from its ability to generate coherent global patterns from simple local rules. Newgas (2020) provides a comprehensive explanation of WFC as a constraint programming approach, where the computer uses built-in algorithms to find solutions to rigorously defined problems rather than following explicit imperative instructions. Recent developments in WFC have extended its applicability to 3D construction scenarios and multiscale generation problems. The integration of WFC with other procedural generation techniques has shown particular promise in creating complex, rule-constrained structures while maintaining computational efficiency.

While construction planning determines placement strategies, accurate structural evaluation requires physics-based models. The integration of physics-based constraints into neural network architectures has revolutionized computational mechanics and structural analysis. Physics-Informed Neural Networks (PINNs) have demonstrated remarkable success in solving partial differential equations while incorporating domain knowledge directly into the learning process Zhang et al. (2024); Bolandi et al. (2023); Kim et al. (2025). Fourier Neural Operators (FNOs) represent a significant advancement in this field, enabling the learning of operators that map between infinite-dimensional function spaces. The application of FNOs to the prediction of the stress-strain field in composite materials has shown exceptional promise Rashid et al. (2022); Khorrami et al. (2024); Shin et al. (2025).

3 Problem Formulation

3.1 Construction Environment

We define our construction environment as a bounded 3D space with coordinates (x, y, z) of size $L \times W \times H$ discretized into unit cells, as illustrated in Figure 1. The construction process involves the sequential placement of LEGO-style bricks with fixed dimensions (l_b, w_b, h_b) in our base configuration. Each placement must satisfy fundamental constraints:

1. **Support constraint**: bricks at height $z > 1$ must be supported by at least one existing brick.
2. **Non-overlap constraint**: new bricks cannot occupy space already filled by existing bricks.
3. **Boundary constraint**: all bricks must lie entirely within the construction space.
4. **Structural Integrity**: ensure that the structure forms a connected component.

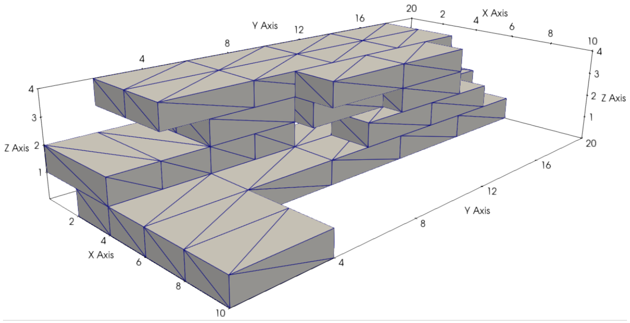


Figure 1: A valid construction. The configuration of $4 \times 2 \times 1$ bricks satisfies all support, non-overlap, boundary, and connectivity constraints.

The scene is represented as a binary occupancy grid of size $L \times W \times H$, where each cell contains a value of 1 if it is occupied by a brick and 0 otherwise. Due to the hierarchical nature of our approach, the state and action spaces for the high- and low-level agents are distinct and are described in detail in Sections 4.1 and 4.2, respectively.

3.2 Optimization Objective and Reward

Unlike previous work Chung et al. (2021); Ahn et al. (2024) that optimizes for shape similarity, we formulate construction as a multiobjective optimization problem. The primary objective is to identify a valid structure s that minimizes von Mises stress:

$$\min_{s \in \Omega} \sigma(s), \quad \text{where } \sigma(s) = \max_{\text{brick} \in s} \sigma_{vm}(s, \text{brick}).$$

Here, Ω denotes the set of structures satisfying the constraints defined earlier in 3.1. $\sigma_{vm}(s, \text{brick})$ is the von Mises stress of a brick in structure s—a single number that measures how large the mechanical stress is in that brick; lower values indicate a safer structure. We describe the computation of this stress using FEM in 3.3, 4.3. Our secondary objectives include achieving the target height h_{target} and minimizing material usage (resource efficiency).

We compute the reward after the complete structure is built (episode end) by running the stress evaluation on the final assembly; intermediate steps receive zero reward. Structural integrity is enforced by the use of action masking. Other objectives are optimized due to maximization of the combined reward function:

$$R_\sigma = \begin{cases} 1, & \text{if } \big(\sigma(s) < \sigma_{\text{thld}}\big) \wedge \big(h = h_{\text{target}}\big) \\ 0, & h < h_{target} \\ 0.5 \cdot \exp\left(\dfrac{\sigma_{\text{thld}} - \sigma(s)}{\eta \cdot \sigma_{\text{thld}}}\right) & \text{otherwise} \end{cases} \tag{1}$$

$$R_h = w_h \cdot \max\left(0, \frac{h_{\text{target}} - |h_{\text{target}} - h|}{h_{\text{target}}}\right) \tag{2}$$

$$R_{\text{vol}} = \begin{cases} w_{vol} \cdot \dfrac{V_{\max} - N_{\text{brick}}}{V_{\max} - h_{\text{target}}}, & \text{if } \big(\sigma(s) < \sigma_{\text{thld}}\big) \wedge \big(h = h_{\text{target}}\big) \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

$$R = \begin{cases} 1 + R_{\text{vol}}, & \text{if } \big(\sigma(s) < \sigma_{\text{thld}}\big) \wedge \big(h = h_{\text{target}}\big) \\ R_\sigma + R_h, & \text{otherwise} \end{cases} \tag{4}$$

where σ_{thld} is the material-dependent stress limit, h is the current structure height, η controls the sharpness of the reward functions for structures that exceed the stress threshold, w_{vol} is a weight for volume-based reward component, w_h balances the height objective against other, N_{brick} is the number of placed bricks in the structure, V_{max} is the maximum allowed number of bricks.

The design of the reward function is essential to guide the agent toward desirable solutions. Our parameter choices are guided by a two-stage learning strategy: first, achieve structural viability, and second, optimize for resource efficiency.

The stress threshold σ_{thld} defines the boundary for a structurally sound design. This threshold is derived from the properties of the brick material and any structure whose maximum von Mises stress exceeds this value is considered failure. The height and stress rewards (R_h, R_σ) reflect the primary goal of training an agent that can successfully build a structure of the target height while meeting the stress constraint. The reward for any "unsuccessful" state (either $h < h_{target}$ or $\sigma >= \sigma_{thld}$) is structured to be strictly less than the reward for a "successful" state. A successful construction receives a base reward of 1, plus a volume bonus. We set $w_h \leq 0.5$ to ensure that even if an agent builds to the full target height (earning the maximum $R_h = w_h$), but fails the stress test (earning $R_\sigma < 0.5$), the total reward ($R = R_\sigma + R_h = R_\sigma + w_h$) remains definitively below the success reward of 1.0. This gradient strongly incentivizes the agent to first satisfy the hard constraints.

The volume reward R_{vol} acts as a bonus reward, applied only after a successful construction is achieved. It introduces the trade-off between resource economy and structural robustness, encouraging the agent to use the fewest bricks possible while potentially creating solutions that are closer to the stress threshold.

3.3 Stress Evaluation

To evaluate structural stability, we develop a finite element model that simulates realistic loading scenarios for brick assemblies. The model incorporates two primary mechanisms: gravitational body forces applied throughout the volume and a uniform surface traction τ applied to the upper surface to simulate operational loads (e.g., payload). As illustrated in Figure 2, the boundary conditions are defined by fixed displacement constraints at the base ($\mathbf{u} = \mathbf{0}$) to represent foundation support, while all lateral boundaries remain free.

The structural analysis is governed by the linear elasticity equations under static equilibrium conditions:

$$\begin{aligned} &\nabla \cdot \boldsymbol{\sigma} + \rho \mathbf{g} = 0 \\ &\boldsymbol{\sigma} = \mathbb{C} : \boldsymbol{\epsilon} \\ &\boldsymbol{\epsilon} = \frac{1}{2}\left(\nabla \mathbf{u} + (\nabla \mathbf{u})^T\right) \end{aligned}$$

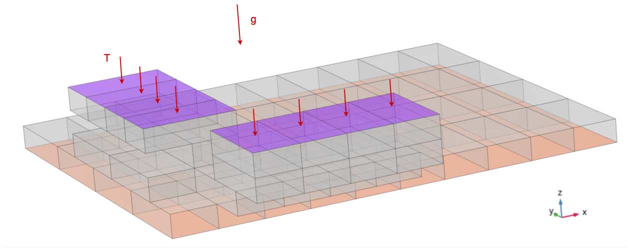


Figure 2: Boundary conditions.

The von Mises stress is given by:

$$\sigma_{vm} = \sqrt{\frac{(\sigma_1 - \sigma_2)^2 + (\sigma_2 - \sigma_3)^2 + (\sigma_3 - \sigma_1)^2}{2}}$$

where $\boldsymbol{\sigma}$ represents the stress tensor, ρ is material density, $\mathbf{g}$ is the gravitational acceleration vector, $\mathbb{C}$ is the fourth-order elasticity tensor, $\boldsymbol{\epsilon}$ denotes the strain tensor, and $\mathbf{u}$ is the displacement field. The constitutive relationship assumes isotropic linear elastic behavior with material properties.

4 Proposed Method

Our training is a two-stage cycle of data collection and policy updates, where each episode produces a complete candidate structure. We consider two planner modes. The 2D planner operates layer-by-layer, proposing 2D rectangular shapes within the current layer at each step, whereas the 3D planner operates on the full 3D scene and proposes 3D shapes without requiring sequential layer planning. The WFC executor then executes this plan layer-by-layer, and the implementation may differ from the proposed plan due to local constraint conflicts. At the end of the episode, the physics module evaluates the structure to compute stress values, which determine the reward.

4.1 High-Level Planner

Operates on a coarse binary occupancy grid $\tilde{s} \in \{0,1\}^{\tilde{L} \times \tilde{W} \times H}$. For the 2D planner, the observation concatenates the downsampled occupancy of the full scene and the current coarse plan for the next layer, which is updated after each action. For the 3D planner, the observation is the coarse plan array alone. In our experiments, $(L, W, H) = (20, 20, 4)$ and $(\tilde{L}, \tilde{W}, H) = (10, 10, 4)$.

We intentionally define the planner's action space on the same coarse grid, so each action corresponds to a large shape that can cover multiple bricks; this biases the high-level policy towards strategic decisions rather than brick-level placement. The planner produces a coarse plan that marks which coarse cells should be filled with bricks. This plan is then upsampled to the original resolution to guide the low-level WFC executor, which operates only on the fine grid. The policy is trained with PPO using physics-based episode-level rewards (1)–(4).

We use the actor–critic architecture, which is illustrated in Figure 3. At each step, the policy selects a staged action $a = (a_{color}, a_{shape}, a_{center})$. A shared 3D convolutional encoder processes the 3D scene, and the resulting features are passed through an MLP to produce actor features. Using actor features, the policy first samples the block type $a_{color} \in \{0, 1\}$; if $a_{color} = 1$, the selected shape region is marked as a usable space for bricks; otherwise, the region is treated as empty (no bricks placed). The policy then samples a shape index a_{shape} from a masked categorical distribution; the shape logits are computed from a concatenation of the actor features and a learned projection of the color logits. Finally, the policy samples the placement center from a distribution whose logits are computed from a concatenation of the actor features, the color embedding, and the shape logits. See Appendix A (Tables 3 and 4) for full architectures.

Feasibility constraints are enforced via action masking: for components a_{shape} and a_{center}, a boolean mask is computed from the current scene state (for a_{center}: also from sampled a_{shape}) and applied by a masked

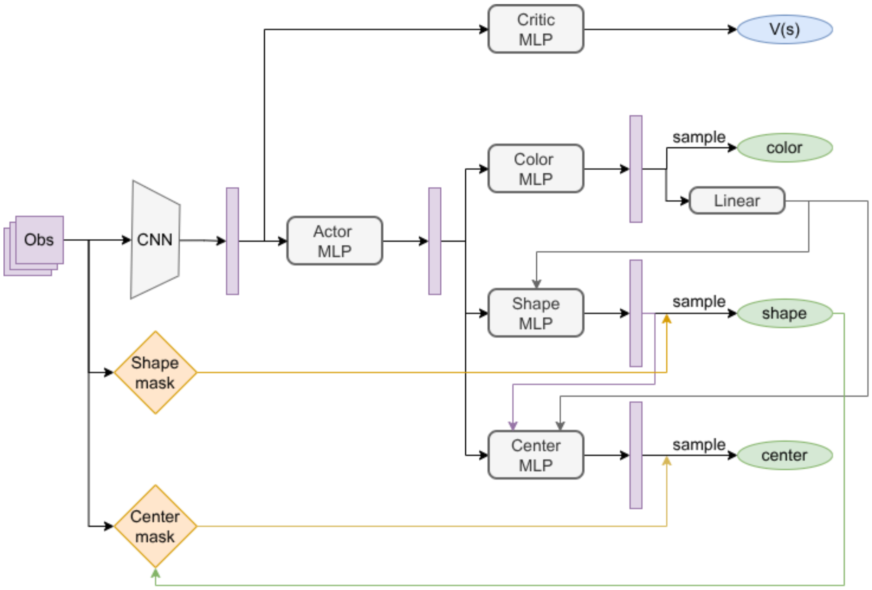


Figure 3: Actor–critic architecture for the high-level planner; the multi-head actor produces a_{color}, a_{shape}, and a_{center} outputs with sequential conditioning.

categorical distribution. a_{shape} is an index in a finite library of rectangular shapes whose side lengths are multiples of the base brick footprint (e.g., for a brick 4×2: 4×2 and 2×4 for a single brick; 4×4 for two bricks; 4×6 for three horizontally aligned bricks). The center action is not selected from the full grid $\{1, \ldots, L\} \times \{1, \ldots, W\}$, instead it selects from a set of valid candidate centers, so the boundary coordinates (e.g. (1,1)) may be invalid because the shape would exceed the scene.

An episode terminates when no feasible placement remains: given the current occupancy grid and the predefined shape library, the action mask becomes empty (i.e., no shape fits in any free region without violating constraints).

4.2 Low-Level WFC Executor

The low-level execution is performed using a modified WFC algorithm specialized for brick assembly. Operates on a fine grid, which is divided into frames of fixed size $F \times F$. The action space consists of $F \times F \times 2 - 1$ discrete actions, representing the (x, y) position and orientation (0 or 1) of each brick. The primary responsibility of the WFC executor is to handle local constraint satisfaction while maximizing the Intersection over Union (IoU) with the high-level strategic plan.

4.3 Physics Integration

The physical validity of structures is verified using a finite element solver. Stress analyses were performed asynchronously through a custom cloud service employing DOLFINx (FEniCS project). This asynchronous approach enabled parallel processing of multiple structural configurations, significantly reducing computational overhead during reinforcement learning training. For additional validation and benchmarking, selected configurations were also analyzed using COMSOL Multiphysics to verify consistency of predictions across different solvers. Mesh resolution was chosen to balance computational efficiency with accuracy, prioritizing rapid feedback and scalability over high-fidelity resolution.

However, a major computational bottleneck arises from relying solely on traditional FEM solvers for the reward calculation in a reinforcement learning loop. To overcome this, we implement a physics approximator based on FNOs, which provides rapid and accurate stress field predictions from geometric data, enabling efficient and scalable training.

Our physics approximator leverages a 3D FNO architecture tailored to predict stress fields from signed distance function (SDF) representations of the built structures. The model processes a single-channel 3D SDF input through a multi-layer FNO core with a fixed number of Fourier modes and hidden channels, followed by a projection head that outputs a single-channel von Mises stress field over the domain.

The trained FNO model replaces the traditional FEM solver in the RL environment's reward pipeline:

1. Convert the current assembly state into its SDF representation.
2. Use the FNO to predict the full 3D von-Mises stress field.
3. Extract stress metrics from the prediction to calculate the reward.

This approach enables the integration of realistic physics into reinforcement learning for structural assembly, where computational efficiency is crucial without compromising fidelity. It allows the RL agent to learn to build minimal-stress 3D structures from bricks efficiently.

5 Experiments

The implementation utilizes JAX/Flax for efficient GPU acceleration and automatic differentiation.

5.1 Experimental Setup

We train the high-level planner with PPO; the full set of PPO hyperparameters is given in Appendix A / Table 2.

The environment is configured as follows:

- **Grids:** High-level agent operates on a $10 \times 10 \times 4$ grid. Low-level agents act within $20 \times 20 \times 4$ local regions.
- **Target height:** 4 units.
- **Episode length:** Up to 25 steps.
- **Parallelization:** 20 concurrent training environments for improved sample efficiency.
- **Reward:** $w_{vol} = 0.5$, $w_h = 0.5$.

For physics simulation, we use realistic mechanics via finite element analysis with a Dolfinx-based solver:

- $\sigma_{thld} = 4 \times 10^8$ Pa.
- **Material:** Aluminum (Young's modulus $E = 68$ GPa, Poisson's ratio $\nu = 0.30$, density $\rho = 2698.9$ kg/m^3).
- **Applied force:** 3×10^6 N uniformly distributed.
- **Mesh parameters:** Factor 0.7.

We set $\sigma_{thld} = 4 \times 10^8$ Pa (400 MPa) as our baseline, corresponding to a conservative yield strength for high-strength aluminum alloys. To demonstrate the physical range of the problem, lower-strength structural steel typically has a yield of 250 MPa, while the ABS plastic has a yield strength of approximately 40–45 MPa.

This setup enables systematic investigation of how hierarchical learning architectures can solve physically constrained engineering problems, requiring an interplay between high-level strategy and low-level execution.

We compare our method to a simplified baseline based on a 2D high-level planner. This baseline restricts high-level decisions to two spatial dimensions: at each height layer, it plans the placement of components without reasoning about full 3D connectivity between layers. The rest of the pipeline—including low-level brick placement using WFC, constraint enforcement, and physics-based reward functions—remains identical. This controlled comparison directly isolates the benefits of full spatial (3D) reasoning for structural stability, efficient material usage, and stress minimization.

5.2 Neural Operator Training

The FNO approximator achieves an 85× to 150× speedup speedup over conventional FEM while maintaining efficient memory usage for large-scale 3D geometric inputs. Training data consists of paired SDF and FEM-derived stress maps from a diverse set of brick assembly configurations:

- Geometric diversity: the structural arrangements range from 20 to 200 bricks in various spatial layouts.
- Stress classification: assemblies are designated as "good" if their stress is below $4 \cdot 10^8$ Pa, whereas those exceeding this limit are considered as "bad," guiding the learning toward safe designs.
- Stratified sampling: guarantees a balanced representation across different brick counts and stress categories.
- Data preprocessing: stress values are logarithmically scaled to stabilize training.

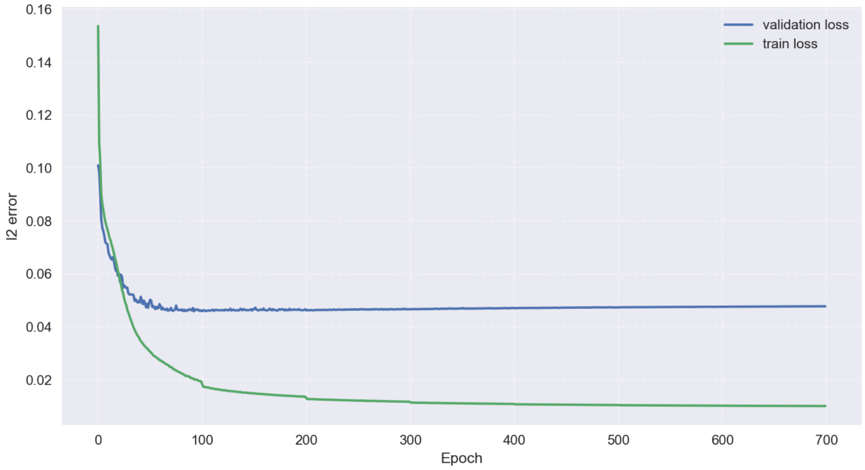


Figure 4: Convergence of the FNO for stress prediction.

The FNO approximator is configured with 16 Fourier modes per spatial dimension to capture the relevant stress gradients across the $64 \times 64 \times 12$ discretization. All architectural and optimization hyperparameters are listed in Appendix A / Table 5. The training performance is illustrated in Figure 4. The plot omits the initial value to prevent a large drop in error (from ~ 1.5 to ~ 0.15) from compressing the y-axis and causing the subsequent convergence to appear negligible. The model validation loss converges rapidly, achieving a minimum of approximately 5% l2-error within the first 80 training epochs. This fast convergence leads to a model that accurately estimates the stress fields in a diverse validation set, which was constructed using stratified sampling to ensure a rich mix of geometric configurations and stress profiles.

Qualitatively, the predicted and true values are very similar: both exhibit negligible stress values in the first and second stages, with maximum stresses concentrated on the upper floor (see Figures 5 and 6). The only notable difference is that the true high-stress values form two distinct clusters, whereas the neural operator predicts a larger contiguous high-stress area that includes both clusters, as well as some space between and around them. Quantitatively, the values are also close: the maximum predicted stress is 3.2×10^9, while the maximum true stress is 3.5×10^9. This corresponds to a relative error of 8.6% in predicting maximum stress, which is acceptable for the calculation of rewards in the reinforcement learning algorithm.

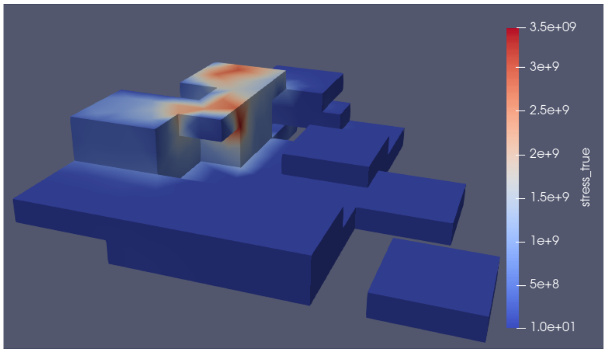


Figure 5: Von Mises stress values computed by the FEM solver

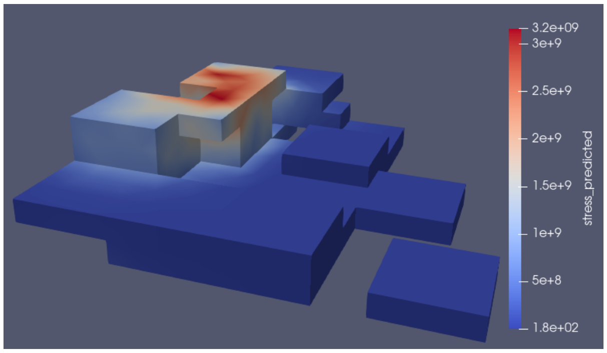


Figure 6: Predicted von Mises stress values by the neural operator

5.3 Structural Performance Results

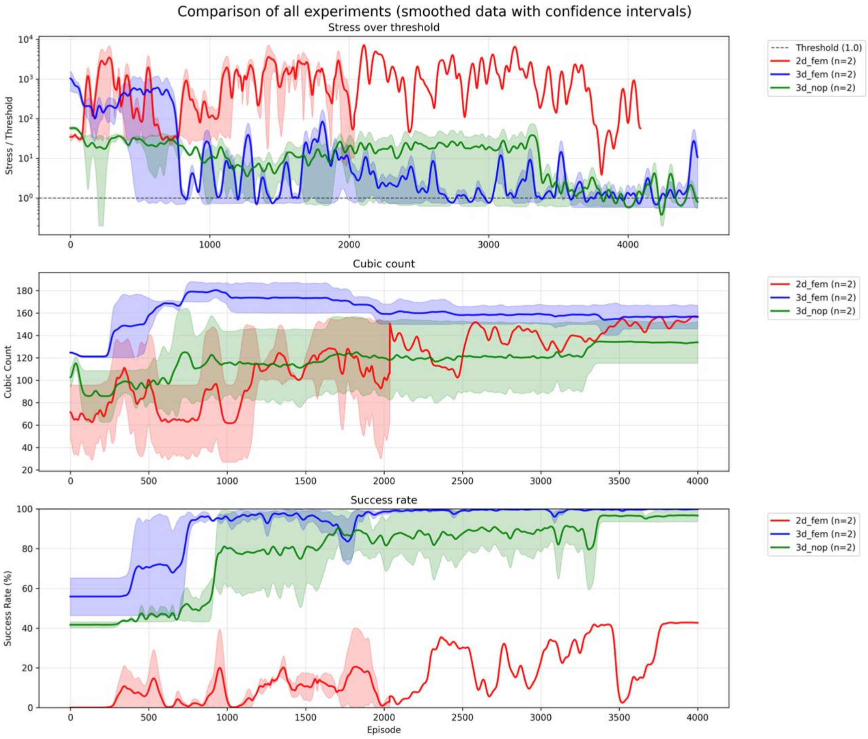


Figure 7: Performance comparison of planner configurations. The plot displays training metrics for three agents: a 2D planner using FEM (red), a 3D planner with FEM (blue), and a 3D planner utilizing the FNO (green).

Figure 7 presents the main quantitative results of our experiments, comparing the performance of our proposed 3D planner (with the FEM and the FNO approximator) against the 2D planner baseline in three key metrics: structural stability, resource efficiency, and overall construction success. The solid lines represent the mean performance averaged in three independent runs with different random seeds. The shaded areas show the corresponding minimum and maximum values of those runs, illustrating the performance variance.

Structural stability (stress over threshold). The primary indicator of a successful design is the stress-to-threshold ratio, shown in the top plot. A value below 1.0 signifies that the maximum von Mises stress is within the acceptable material limits. As illustrated, both 3D planners successfully learn policies that reduce the stress-to-threshold ratio below 1.0, whereas the 2D planner consistently fails to do so. A closer comparison of the 3D planners reveals that the agent guided by the FEM solver exhibits faster convergence, achieving acceptable performance in fewer than 3,000 PPO iterations. However, the FNO-based agent also reliably converges to the required performance level.

Overall performance (success rate). The bottom plot provides the percentage of episodes in a batch that result in a complete structure (that is, the target height is met and the stress-to-threshold ratio is below 1.0). This metric offers a decisive look at overall performance. Both 3D planners (blue and green) converge to near-perfect success rates, with the operator-based agent requiring more interactions to do so. In contrast, the 2D planner exhibits strong fluctuations and peaks at only about 40%, underscoring its instability and limited effectiveness.

Resource efficiency. The middle plot shows the number of bricks used by each agent. This metric reveals the trade-off between the resource economy and structural integrity. Although using fewer bricks is efficient, building a stable, full-height structure is difficult with too few materials. In contrast, a high brick count does not guarantee low stress.

The plot shows that all agents initially increase the brick count to expand the load-bearing area at the top layer, which lowers the stress-to-threshold ratio by distributing loads more evenly. The 2D planner demonstrates an upward trend across training, which is consistent with its low success rate: because unsuccessful episodes do not receive the volume bonus, there is little incentive to reduce brick usage, while the stress-driven component still rewards reductions in peak stress—often achieved by adding bricks. By contrast, the 3D planner's brick count (blue) begins to decline after roughly 1,000 iterations, coinciding with near-perfect success rates; this situation indicates that the volume bonus starts to dominate the optimization, guiding the agent to reduce material while maintaining structural feasibility.

Taken together, these results highlight the clear superiority of the hierarchical 3D planning approach. Furthermore, they provide strong evidence that FNO can serve as a high-fidelity, computationally efficient substitute for traditional FEM solvers within the reinforcement learning loop, enabling effective and scalable training for complex, physics-constrained tasks.

State space complexity. The state space in our construction environment scales exponentially with grid resolution. For a $20 \times 20 \times 4$ voxel grid with binary occupancy states, the theoretical state space encompasses $2^{1600} \approx 10^{482}$ possible configurations. When constrained to valid $4 \times 2 \times 1$ brick placements, each brick admits 2,584 distinct placement positions, yielding combinatorial complexity that renders exhaustive search intractable.

Our hierarchical agent architecture successfully identifies structurally sound, material-efficient solutions within this space. The key insight lies in the emergence of implicit construction heuristics not explicitly programmed into the reward function:

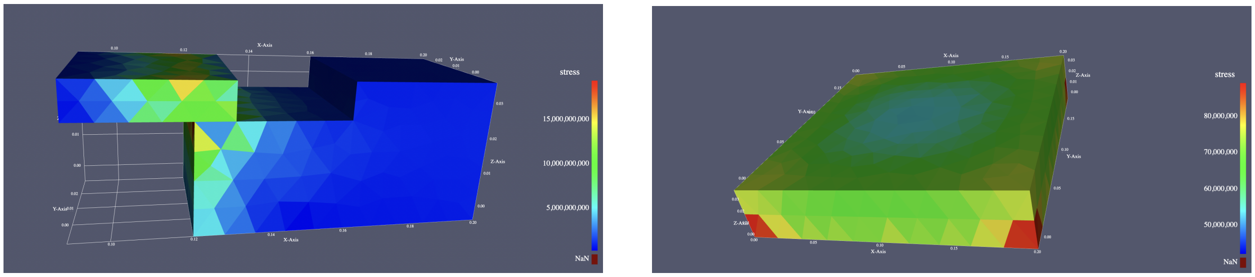
- **Structural coherence**: Avoiding unsupported elements without explicit geometric constraints.
- **Load distribution**: Utilizing symmetrical configurations to minimize stress concentrations.

5.4 Qualitative Analysis

To further elucidate the learning dynamics and qualitative outcomes of our approach, we provide visual examples of structures generated at different stages of training, focusing on stress distributions and structural characteristics.

During the initial training phase, the model frequently generated suboptimal configurations (Figure 8a), including designs with hanging or poorly supported bricks. These structures were unable to meet the allowable stress criteria, as evidenced by substantial stress concentrations that exceeded the material threshold.

As training progressed, the agent learned to construct more stable assemblies (Figure 8b). The frequency of failure cases decreased and the stress distribution became more uniform, although in certain regions local peaks in stress still appeared. These intermediate solutions often met the target height and connectivity requirements, but did not optimize material efficiency yet.

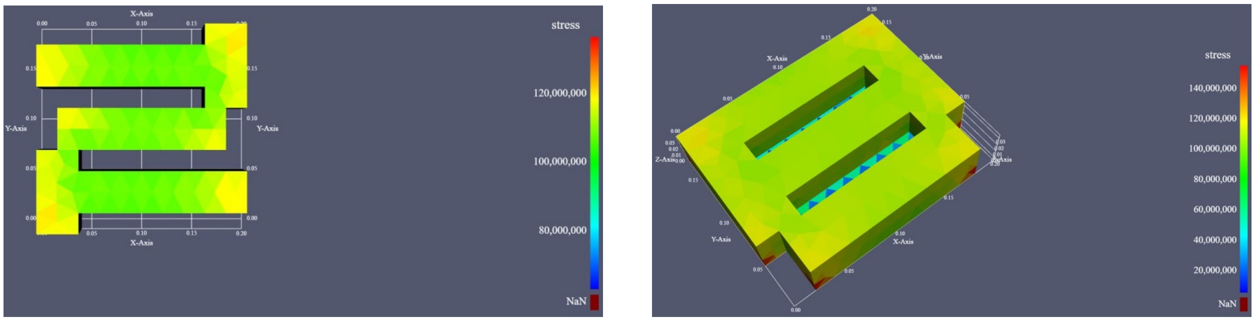


(a) Such designs often failed to satisfy the prescribed stress threshold and frequently contained overhanging elements with inadequate support.

(b) The stress is more evenly distributed, with the majority of the structure satisfying the threshold, though isolated high-stress regions remain.

Figure 8: Early and intermediate-stage structures generated during training. The left panel illustrates a failure case with high stress concentrations, while the right panel shows a partially stabilized structure where most stresses remain below the critical threshold, although some localized peaks persist.

In the later stages of training, the model consistently produces high-quality structures (Figures 9a and 9b). These solutions not only conform to the target stress limits, but also significantly reduce the number of bricks required, demonstrating the efficiency of the learned material without sacrificing structural integrity. The stress gradients are smoother and critical regions are reinforced, reflecting advanced planning and a refined balance between stability and resource usage.



(a) Mature design after extended training. The structure is compact, uses fewer bricks, and successfully maintains stresses below the prescribed limit. This demonstrates the agent's ability to discover both efficient and physically valid solutions.

(b) Alternative top-down view of an optimized design at convergence. The agent has learned to maximize spatial efficiency while ensuring all stress regions remain safely below the failure threshold.

Figure 9: Mature structures achieved after extended training. Both designs meet the stress constraints while using significantly fewer blocks. The solutions demonstrate the agent's ability to optimize both material efficiency and structural reliability.

In general, these results illustrate a key property of our approach: the ability to autonomously discover design strategies that satisfy physical requirements and optimize resource consumption. The qualitative progression from failure cases to threshold-compliant solutions to resource-effective constructions demonstrates that the proposed framework is capable of sophisticated, constraint-aware reasoning in complex structural assembly tasks.

Our results highlight that hierarchical decomposition is central to effective structural construction: separating strategic planning from tactical execution improves structural performance. The high-level planner can focus on global optimization, while the low-level executor ensures local constraint satisfaction. Despite the apparent simplicity of our discrete assembly task, the approach demonstrates effective navigation of a large solution space. The hierarchical learning architecture enables the agent to discover construction strategies and structural principles that extend beyond the explicitly encoded reward signals. Incorporating stress analysis directly into the reward function enables the learning of structurally sound construction strategies that would be difficult to discover through shape-based objectives alone, while the WFC-based executor provides efficient constraint handling, making real-time construction with complex constraint sets feasible.

6 Conclusion

This study demonstrates that hierarchical decomposition, paired with physics-integrated rewards, enables agents to discover structures that satisfy stress constraints and minimize material, surpassing a 2D planning alternative that lacks full spatial reasoning. The low-level WFC executor complements the high-level policy by enforcing local feasibility and alignment with the global plan, supporting consistent success across runs and seeds. Moreover, an FNO-based surrogate effectively replaces FEM for reward computation without degrading ultimate performance, offering a practical path to scalable training in physics-constrained design tasks.

Although the framework demonstrates robust structural learning, several limitations remain. Physics simulation introduces computational overhead, though asynchronous implementation and neural operator surrogates partially mitigate this impact. The current implementation uses simplified material models and loading conditions, with a planned extension to more complex structural analysis for future work. Finally, experiments focus on a single brick type; extension to multiple brick types and complex shapes is ongoing.

References

Seokjun Ahn, Jungtaek Kim, Minsu Cho, and Jaesik Park. Budget-aware sequential brick assembly with efficient constraint satisfaction. *Transactions on Machine Learning Research*, 2024.

Hamed Bolandi, Gautam Sreekumar, Xuyang Li, Nizar Lajnef, and Vishnu Naresh Boddeti. Physics informed neural network for dynamic stress prediction. *Appl. Intell.*, 53(22):26313–26328, 2023.

Hyunsoo Chung, Jungtaek Kim, Boris Knyazev, Jinhwi Lee, Graham W Taylor, Jaesik Park, and Minsu Cho. Brick-by-brick: Combinatorial construction with deep reinforcement learning. *Advances in Neural Information Processing Systems*, 34:5745–5757, 2021.

Maxim Gumin. Wave function collapse, 2016. URL https://github.com/mxgmn/WaveFunctionCollapse.

Reza Kakooee and Benjamin Dillenburger. Reimagining space layout design through deep reinforcement learning. *Journal of Computational Design and Engineering*, 11(3):43–55, 2024.

Mohammad S Khorrami, Pawan Goyal, Jaber R Mianroodi, Bob Svendsen, Peter Benner, and Dierk Raabe. A physics-encoded fourier neural operator approach for surrogate modeling of divergence-free stress fields in solids. *arXiv preprint arXiv:2408.15408*, 2024.

Kwang Hyeon Kim, Hae-Won Koo, and Byung-Jou Lee. Predictive stress analysis in simplified spinal disc model using physics-informed neural networks. *Computer Methods in Biomechanics and Biomedical Engineering*, pp. 1–13, 2025.

Maksim Kolodiazhnyi, Denis Tarasov, Dmitrii Zhemchuzhnikov, Alexander Nikulin, Ilya Zisman, Anna Vorontsova, Anton Konushin, Vladislav Kurenkov, and Danila Rukhovich. cadrille: Multi-modal cad reconstruction with online reinforcement learning. *arXiv preprint arXiv:2505.22914*, 2025.

Adam Newgas. Wave function collapse tips and tricks, 2020. URL https://www.boristhebrave.com/2020/02/08/wave-function-collapse-tips-and-tricks/.

Meer Mehran Rashid, Tanu Pittie, Souvik Chakraborty, and NM Anoop Krishnan. Learning the stress-strain fields in digital composites using fourier neural operator. *Iscience*, 25(11), 2022.

Minwoo Shin, Minjee Seo, Hayoung Choi, Jaemin Jung, and Kyungho Yoon. Local stress fields prediction using global displacement through fourier neural operators. *Journal of Computational Design and Engineering*, 12(5):21–40, 2025.

Wenhao Zhang, Pinghe Ni, Mi Zhao, and Xiuli Du. A general method for solving differential equations of motion using physics-informed neural networks. *Applied Sciences*, 14(17):7694, 2024.

A Implementation Details

The high-level PPO planner employs a 3D convolutional neural network (CNN) encoder to process the 3D grid-based environment. The encoder consists of a two-layer 3D CNN with ReLU activations: the first convolution layer maps from 1 input channel (binary occupancy) to 32 channels using a 3×3×3 kernel, followed by a second layer transforming 32 channels to 16 output channels with the same kernel size. Padding and stride settings maintain the spatial resolution throughout the encoding. The encoder thus preserves spatial structure and encodes volumetric features relevant for planning.

Parameter	**Value**
coarse grid size	10
fine grid size	20
target height	4
num_envs	20
use augmentation	True
wfc greedy collapse	True

Table 1: Main hyper-parameters.

Parameter	**Value**
total_timesteps	1,000,000
buffer_size	5000
num_episodes_in_batch	20
anneal_lr	True
update_epochs	10
tau	0.95
norm_adv	True
clip_coef	0.2
clip_vloss	True
entropy_coeff	0.0015
critic_coeff	0.5
gradient_clip	5.0
target_kl	0.01
num_minibatches	2
learning_rate	0.0002
gamma	0.999

Table 2: PPO hyper-parameters.

Component	**Architecture**
Actor trunk	Linear($d \to 64$) + Tanh; Linear($64 \to 64$) + Tanh
Critic	Linear($d \to 64$) + Tanh; Linear($64 \to 64$) + Tanh; Linear($64 \to 1$)
Policy heads	Linear($64 \to \lvert\mathcal{A}_{color}\rvert$); Linear($64 \to \lvert\mathcal{A}_{shape}\rvert$); Linear($64 \to \lvert\mathcal{A}_{center}\rvert$)

Table 3: Actor–critic architecture (2D planner). Here d is the flattened encoder output dimension.

Component	**Architecture**
Actor trunk	Linear($d \to 256$) + ReLU; Linear($256 \to 256$) + ReLU
Critic	Linear($d \to 256$) + Tanh; Linear($256 \to 256$) + Tanh; Linear($256 \to 1$)
Color head	Linear($256 \to 64$)+ReLU; Linear($64 \to 64$)+ReLU; Linear($64 \to \lvert\mathcal{A}_{color}\rvert$)
Shape head	Linear($256 + 64 \to 64$)+Tanh; Linear($64 \to 64$)+Tanh; Linear($64 \to \lvert\mathcal{A}_{shape}\rvert$)
Center head	Linear($256 + 64 + \lvert\mathcal{A}_{shape}\rvert \to 128$)+Tanh; Linear($128 \to 128$)+Tanh; Linear($128 \to \lvert\mathcal{A}_{center}\rvert$)

Table 4: Actor–critic architecture (3D planner).

Parameter	**Value / Setting**
Input resolution	$64 \times 64 \times 12$
Input channels	1 (SDF)
FNO layers	4
Fourier modes (per spatial dim.)	16
Hidden channels	32
Projection channel ratio	4
Output channels	1 (von Mises stress)
Optimizer	AdamW
Learning rate	3×10^{-4}
Weight decay	1×10^{-3}
LR schedule	Cosine annealing
Training epochs	700

Table 5: Hyperparameters for the FNO-based physics approximator.