Neural Modular World Model

Minghao Guo[1] **Zhiyang Dou**[1 2] **Chong Zeng**[3 4] **Wojciech Matusik**[1]

Abstract

We introduce the Neural Modular World Model, a graphics-inspired framework that decomposes learned world models into two interoperable layers: *Attribute modules* infer scene-specific descriptors—geometry, material, appearance, and illumination—via either explicit assets or compact neural fields. *Function modules* supply reusable, content-agnostic operators for physics integration and image synthesis. This attribute–function split combines the expressive power of large generative networks with the physical guarantees and interpretability of classical simulation, yielding (i) photorealistic and physically accurate predictions, (ii) fine-grained controllability through independent module dials, and (iii) precise error attribution for system identification and targeted debugging. We anticipate that the modular structure will unlock orthogonal scaling, letting geometry, physics, and rendering grow independently, rather than balloon together in a single monolithic network. These properties make Neural Modular World Model a practical blueprint for building planet-scale, physically faithful world models.

1. Introduction

World models (Ha & Schmidhuber, 2018; Forrester, 1971; Grzeszczuk et al., 1998)—generative predictors of how percepts evolve as an agent acts—now underpin a wide range of applications, including planning in robotics (Yang et al., 2023; Wu et al., 2023; Zhou et al., 2024), autonomous driving (Hu et al., 2023; Jia et al., 2023), and simulated agent control (Grzeszczuk et al., 1998; Hansen et al., 2024; Hafner et al., 2019; Fussell et al., 2021), among others. Early work distilled Atari games into compact recurrent states, and later video-transformer models scaled prediction to natural, first-person streams. Yet these systems still falter on tasks that require fine-grained physical fidelity, e.g., pushing deformable objects or maintaining long-range coherence; a truly plausible world model must also encode the causal dynamics of matter, light, and interaction.

Conventional world models are trained end-to-end on raw pixels with generic likelihood or prediction losses and minimal inductive bias. They implicitly intertwine three vastly different spaces: high-dimensional appearance, low-dimensional latent physics, and control, into a single monolithic network. This entanglement forces the learner to discover latent physical structure (geometries, materials, conservation laws, etc.) purely from image or video statistics. This intractably large search space yields brittle generalization and spurious correlations. In effect, physics is treated as a *hidden* variable to be backed out from observation alone, rather than an *explicit* scaffold guiding representation.

Inspired by studies of cortical modularity (Lake et al., 2017) and by compositional networks in scene understanding (Andreas et al., 2016), we argue that a world model should itself be *modular*. Decomposing along interpretable factors (geometry, appearance, dynamics) isolates independent priors, facilitates data reuse, and allows targeted supervision where ground-truth is available (e.g., depth, force sensors, motion capture). Modular design also mirrors the software engineering practice of physics engines and rendering pipelines, suggesting a natural path to scalable, testable components.

But *what* modules, and *how* should they communicate? Computer graphics offers a pragmatic blueprint. Graphics pipelines routinely achieve two outcomes that set the bar for learned models: they render photorealistic imagery—frames virtually indistinguishable from real photographs—while simultaneously guaranteeing physically accurate motion, faithfully obeying the laws of mechanics as objects deform, collide, and interact with light. Decades of production rendering show that scenes can be factorized into *attributes* (shape, material, illumination) queried by *functions* (integrators, shaders) that enforce analytic physical rules. Crucially, this modularity delivers two concrete advantages for learning systems: (1) **controllability**, because each attribute dial can be turned independently, allowing a user or a downstream planner to morph geometry, swap materials, or relight a scene without retraining the rest of the network; and (2) **error attribution**, because any deviation between prediction and reality can be traced to the specific module

[1] MIT CSAIL [2] The University of Hong Kong [3] Stanford University [4] Zhejiang University. Correspondence to: Minghao Guo <guomh2014@gmail.com>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

responsible, so one can fine-tune or swap out that component without destabilizing the entire pipeline. Replicating this architecture in a learned system therefore establishes a powerful *empirical lower bound*: if our neural modules respect the same attribute–function contract, the resulting world model should, at minimum, match the fidelity, controllability, and interpretability that classical graphics already delivers—while still gaining the flexibility and data-driven scalability of modern deep learning.

Building on this insight, we introduce the **Neural Modular World Model**, which instantiates two fundamental module classes (Fig.1). **Attribute modules** perform inference: from an input view or text prompt, they regress *explicit* scene attributes, e.g., triangular meshes, spatially varying BRDFs, environment lighting probes, or provide continuous *implicit* fields such as signed-distance functions or neural constitutive laws. Each exposes a simple query–response interface mapping 3D positions, directions, or tetrahedral indices to attribute values for maximal controllability. **Function modules** are *content-agnostic* neural surrogates of traditional graphics operators. Integrator modules advance physical state, while shader modules transform physical interactions into pixel intensities. Because they encode universal laws, e.g., energy conservation, momentum, and radiometry, they are pre-trained once with strong physics-based regularization and reused across tasks without fine-tuning.

This division of labor captures the best of both worlds: the expressive power and scalability of modern deep learning, and the hard guarantees and intuitive controls of analytic simulation. Clear boundaries between attributes and functions ensure that every prediction respects fundamental physical laws while still allowing the model to inherit the richness and diversity of learned assets. As a result, the system produces scenes that look real and behave correctly, satisfying visual and physical constraints in tandem.

2. Neural Modular World Model

Fig. 1 gives an overview of our Neural Modular World Model. An input image or text prompt is first decoded into four *attribute modules*, i.e., geometry, material, appearance, and lighting. These attributes are then consumed by two *function modules*: a neural integrator that predicts physical states and a neural shader that transforms scene states into pixel values. Attribute modules are scene–specific inference networks, whereas function modules are scene-agnoistic reusable neural surrogates of universal physical laws.

2.1. Attribute Modules

Attribute modules infer and represent scene–specific quantities from raw observations and present them through uniform query interfaces. Each module can be instantiated in an *explicit* form, returning traditional graphics assets that are easy to visualize and edit, or an *implicit* form, returning continuous neural fields that trade compactness and differentiability for direct interpretability. The function modules consume these attributes without modification, so users are free to mix and match explicit and implicit variants.

Neural Geometry Module The geometry module describes *what is where*. In its explicit guise, it predicts watertight triangle meshes, tetrahedral meshes, or point clouds that compile into surface descriptions. The implicit alternative is a signed-distance or occupancy network queried with 3D coordinates and returning inside/outside values and normals on the fly. Regardless of representation, each object is emitted in a canonical rest frame together with learned latent codes that summarize local style and mechanical class. This ensures that material and appearance modules can modulate behavior when multiple objects share the same geometry.

Neural Material Module Materials determine *how matter deforms or resists force*. The explicit route regresses analytic constitutive parameters (Young's modulus, Poisson ratio, damping coefficients, etc) that plug directly into FEM or rigid–body simulators. The implicit route learns a small MLP that maps strain tensors to stress responses, effectively replacing hand-crafted constitutive models with data-driven surrogates. Inputs include the latent mechanical code that the geometry module exports, enabling one shared network to cover plastics, elastomers, and granular aggregates.

Neural Appearance Module Appearance specifies *how light interacts with surfaces*. An explicit appearance module outputs spatially varying BRDF or BSSRDF parameters, texture atlases, or microfacet distributions. Implicitly, a neural reflectance field takes position, normal, and view/light direction as input and returns reflected radiance, seamlessly capturing fine-scale detail such as anisotropic fibres or metallic flakes that exceed the capacity of analytic models. Because reflectance is queried per shading point, we can achieve real-time material edits, e.g., switching plastic to brushed aluminium, by simply swapping appearance latents while leaving geometry and lighting untouched.

Neural Lighting Module Lighting describes *where photons come from*. The explicit variant predicts an HDR environment map, sun-sky rig, or a set of spatially extended area lights positioned in 3D space; these outputs can be previewed instantly in any path tracer. The implicit variant is a generative radiance field that, conditioned on a text phrase like "warm indoor cafe at dusk," samples plausible global illumination that conforms to the scene's color palette and desired mood. Crucially, the lighting module is fully plug-and-play: once trained, it can illuminate previously unseen geometry without fine-tuning.

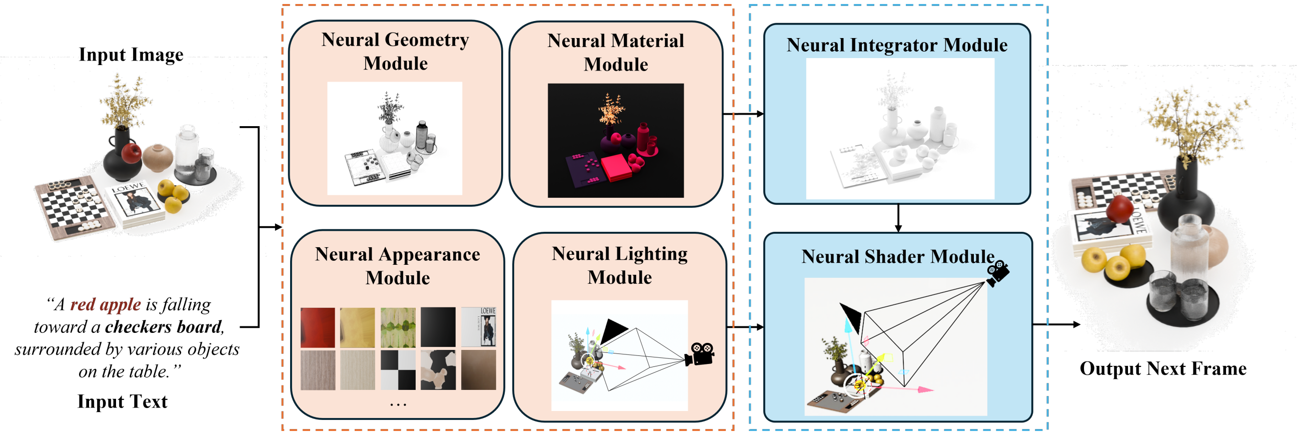

Figure 1. Overall pipeline of the Neural Modular World Model. Beige blocks are attribute (inference) modules; blue blocks are function (physics) modules. Together, they form a graphics-inspired world model that achieve high-fidelity dynamics and markedly more accurate long-horizon state predictions, thanks to the modular design.

2.2. Function Modules

Function modules encode *universal physical laws*: they take attributes produced by the scene-specific inference blocks and transform them with operators that apply everywhere and to every object. Much like a traditional physics engine plus a film renderer, they form the backbone that enforces dynamical and photometric consistency.

Neural Integrator Module. The integrator takes the current physical state of every object—positions, velocities, and latent material codes—and returns the next state after a time step. Because it is a pure mapping from state_t to state_{t+1}, it can be trained entirely on synthetic trajectories generated by any simulator; no real-sensor idiosyncrasies enter the data, so the usual sim-to-real gap is absent.

Neural Shader Module. The shader consumes a transient scene description—rasterized geometry buffers, per-pixel appearance codes, lighting directions, and visibility masks—and outputs the corresponding RGB frame. Like the integrator, it is a self-contained function and can be supervised with rendered imagery produced offline, avoiding real-world capture artifacts. Together, these two modules form a closed "act-and-see" loop: the integrator advances physics, and the shader turns the resulting state into pixels, both learned safely and scalably in simulation.

3. Discussion

Practical implementation. A production-ready prototype can be assembled almost entirely from open-source code. Geometry can be inferred with image-to-mesh networks such as Xiang et al. (2024); He et al. (2025); neural constitutive laws can adopt the graph-network material decoders of Ma et al. (2023); surface appearance can be handled by Sztrajman et al. (2021); illumination can be regressed by environment-map models like Liang et al. (2023); Song & Funkhouser (2019). Scene dynamics may be learned with Pfaff et al. (2020) or supervised directly with differentiable FEM kernels provided by Warp (Macklin, 2022) and Taichi (Hu et al., 2019). Finally, raster-time shading can be supplied by real-time path tracers such as RTXDI.

Swapping explicit modules. Every module can accept either an analytic asset or its neural surrogate. A VFX house can keep hand-sculpted meshes and measured BRDFs yet drop in the neural integrator for cloth dynamics; a robotics lab can hold on to Bullet's rigid-body engine while replacing the shader with a diffusion up-sampler for domain randomization. Because interface contracts never change, these substitutions require no retraining, only a flag flip.

Error attribution, system identification, and inverse inference. Modular gradients let us do more than debug: they enable *system identification*. Pixel residuals propagate to exactly the offending block, e.g., appearance if colors drift, integrator if trajectories diverge, so we can fine-tune one component in situ while freezing the rest. Conversely, we can invert the pipeline: holding the shader and integrator fixed, an optimizer can solve for geometry, material parameters, or lighting that best explain a video sequence, yielding a principled route to data-driven asset creation. Such fine-grained attribution is impossible in undifferentiated video models and becomes a distinctive strength of our design.

Outlook. By marrying the scalability of foundation models with the transparency and interchangeability of classical graphics, the Neural Modular World Model offers a realistic path toward planet-scale, physically faithful simulators that not only *look* right but also *behave* right, and whose parts can be inspected, upgraded, or replaced as engineering advances march on.

References

Andreas, J., Rohrbach, M., Darrell, T., and Klein, D. Neural module networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 39–48, 2016.

Forrester, J. W. Counterintuitive behavior of social systems. *Theory and decision*, 2(2):109–140, 1971.

Fussell, L., Bergamin, K., and Holden, D. Supertrack: Motion tracking for physically simulated characters using supervised learning. *ACM Transactions on Graphics (TOG)*, 40(6):1–13, 2021.

Grzeszczuk, R., Terzopoulos, D., and Hinton, G. Neuroanimator: Fast neural network emulation and control of physics-based models. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pp. 9–20, 1998.

Ha, D. and Schmidhuber, J. World models. *arXiv preprint arXiv:1803.10122*, 2018.

Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

Hansen, N., SV, J., Sobal, V., LeCun, Y., Wang, X., and Su, H. Hierarchical world models as visual whole-body humanoid controllers. *arXiv preprint arXiv:2405.18418*, 2024.

He, X., Zou, Z.-X., Chen, C.-H., Guo, Y.-C., Liang, D., Yuan, C., Ouyang, W., Cao, Y.-P., and Li, Y. Sparseflex: High-resolution and arbitrary-topology 3d shape modeling. *arXiv preprint arXiv:2503.21732*, 2025.

Hu, A., Russell, L., Yeo, H., Murez, Z., Fedoseev, G., Kendall, A., Shotton, J., and Corrado, G. Gaia-1: A generative world model for autonomous driving. *arXiv preprint arXiv:2309.17080*, 2023.

Hu, Y., Li, T.-M., Anderson, L., Ragan-Kelley, J., and Durand, F. Taichi: a language for high-performance computation on spatially sparse data structures. *ACM Transactions on Graphics (TOG)*, 38(6):1–16, 2019.

Jia, F., Mao, W., Liu, Y., Zhao, Y., Wen, Y., Zhang, C., Zhang, X., and Wang, T. Adriver-i: A general world model for autonomous driving. *arXiv preprint arXiv:2311.13549*, 2023.

Lake, B. M., Ullman, T. D., Tenenbaum, J. B., and Gershman, S. J. Building machines that learn and think like people. *Behavioral and brain sciences*, 40:e253, 2017.

Liang, R., Chen, H., Li, C., Chen, F., Panneer, S., and Vijaykumar, N. Envidr: Implicit differentiable renderer with neural environment lighting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 79–89, 2023.

Ma, P., Chen, P. Y., Deng, B., Tenenbaum, J. B., Du, T., Gan, C., and Matusik, W. Learning neural constitutive laws from motion observations for generalizable pde dynamics. In *International Conference on Machine Learning*, pp. 23279–23300. PMLR, 2023.

Macklin, M. Warp: A high-performance python framework for gpu simulation and graphics. https://github.com/nvidia/warp, March 2022. NVIDIA GPU Technology Conference (GTC).

Pfaff, T., Fortunato, M., Sanchez-Gonzalez, A., and Battaglia, P. Learning mesh-based simulation with graph networks. In *International conference on learning representations*, 2020.

Song, S. and Funkhouser, T. Neural illumination: Lighting prediction for indoor environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6918–6926, 2019.

Sztrajman, A., Rainer, G., Ritschel, T., and Weyrich, T. Neural brdf representation and importance sampling. In *Computer Graphics Forum*, volume 40, pp. 332–346. Wiley Online Library, 2021.

Wu, P., Escontrela, A., Hafner, D., Abbeel, P., and Goldberg, K. Daydreamer: World models for physical robot learning. In *Conference on robot learning*, pp. 2226–2240. PMLR, 2023.

Xiang, J., Lv, Z., Xu, S., Deng, Y., Wang, R., Zhang, B., Chen, D., Tong, X., and Yang, J. Structured 3d latents for scalable and versatile 3d generation. *arXiv preprint arXiv:2412.01506*, 2024.

Yang, M., Du, Y., Ghasemipour, K., Tompson, J., Schuurmans, D., and Abbeel, P. Learning interactive real-world simulators. *arXiv preprint arXiv:2310.06114*, 1(2): 6, 2023.

Zhou, S., Du, Y., Chen, J., Li, Y., Yeung, D.-Y., and Gan, C. Robodreamer: Learning compositional world models for robot imagination. *arXiv preprint arXiv:2404.12377*, 2024.