# Neural Hamiltonian Deformation Fields for Dynamic Scene Rendering

Hai-Long Qin[1], Sixian Wang[2], Guo Lu[2], Jincheng Dai[1]

[1]Beijing University of Posts and Telecommunications, [2]Shanghai Jiao Tong University

**Hai-Long Qin**

**December, 2025**

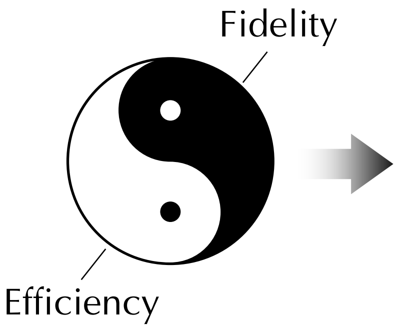
# Background & Motivation

## Dynamic Scene Rendering

- **The Goal:** Model scene dynamics and synthesize high-fidelity novel views at arbitrary timestamps in real-time.

Given Views → (Spatial) → Novel Views → (Temporal) → 4D Video

- **The Challenge:** Fidelity-efficiency trade-off.
- **Existing Methods:** NeRF- and Gaussian Splatting-based deformation fields prediction, struggling to simultaneously satisfy both ends.

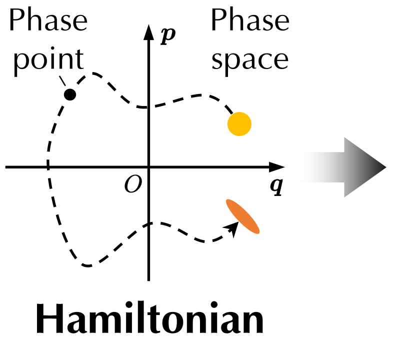



**1. Neural Radiance Field-based**
High rendering quality but slow: dense sampling along rays needed

**2. Gaussian Splatting-based**
Fast and practical, but poor at modeling complex dynamics

## What Feels High-fidelity (Realism)

- **Our Argument:** Realistic rendering requires both perceptual quality and physical plausibility, yet most current methods focus solely on the former.
- **Cognitive Analogy:** Human cognition and scene rendering both follow fundamental physical laws, particularly Hamiltonian mechanics.




**1. Human Cognition**

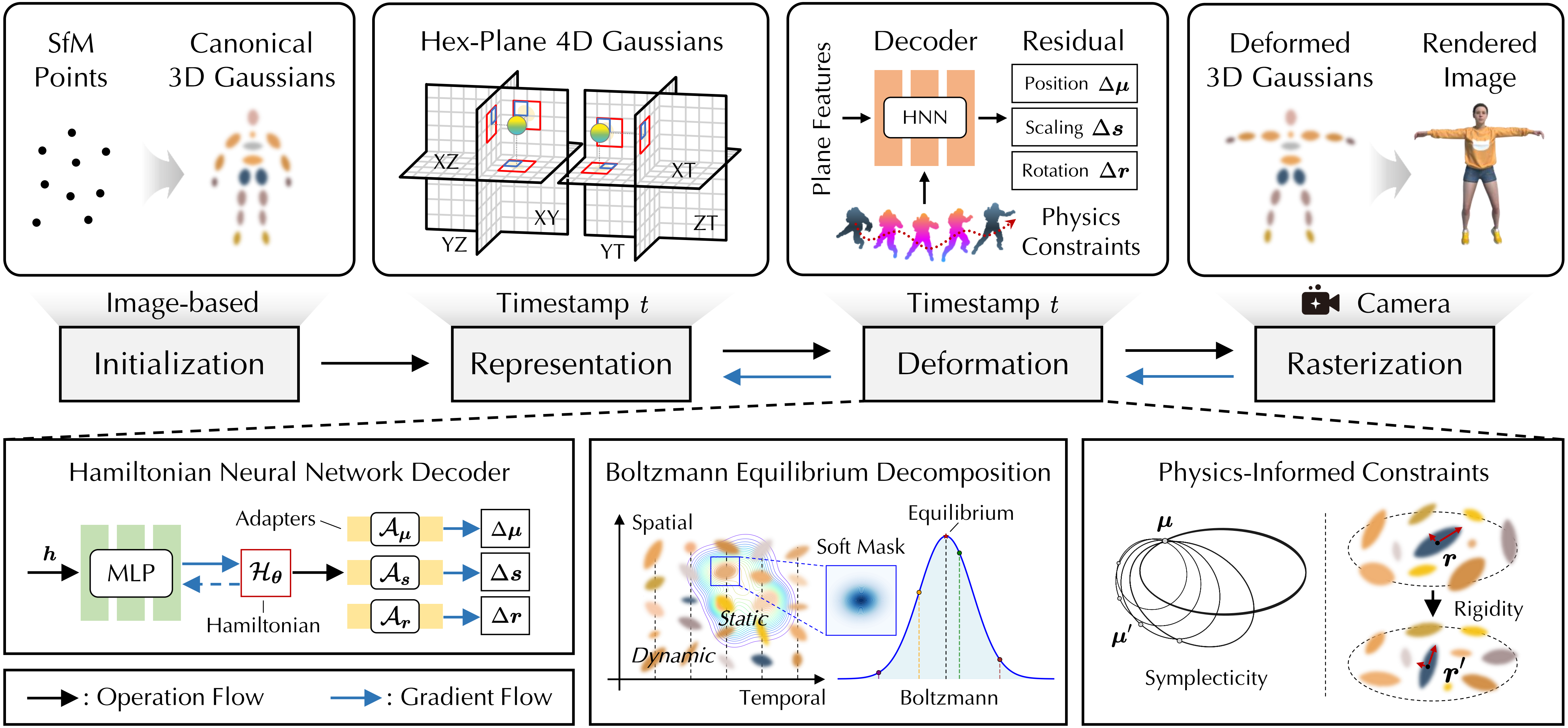
Perception → Abstraction → Prediction

**2. Scene Rendering**

Initialization → Representation → Deformation

- **Core Insight:** Gaussians evolve in a phase space following Hamiltonian mechanics due to their symplectic covariances.

# Overall Pipeline: NeHaD



# Hamiltonian Neural Network Decoder

**Why HNN?** Standard MLPs lack the ability to learn Hamiltonian physics priors.

## Hamiltonian Formulation

- **Energy Conservation:** $\mathcal{H}(\boldsymbol{q}, \boldsymbol{p}) = \mathcal{U}(\boldsymbol{q}) + \mathcal{K}(\boldsymbol{p})$
- **Canonical Equations:** $\dot{\boldsymbol{q}} = \partial \mathcal{H} / \partial \boldsymbol{p}, \quad \dot{\boldsymbol{p}} = -\partial \mathcal{H} / \partial \boldsymbol{q}$

**1. Explicit Position-Momentum Coupling As Input**
Intractable to high-dimensional Gaussian primitives

**2. Vector Fields As Learning Objective**
Dimensionality curse, energy conservation not guaranteed

## Our Implementation

- **Implicit Features As HNN Input:** Extract spatial-temporal hex-plane features and map into latent $\boldsymbol{h}$ using an MLP.
- **Decomposed Vector Fields Learned by Scalar Potentials:** Equivalently learning Hamiltonian vector fields via $F_1, F_2$.

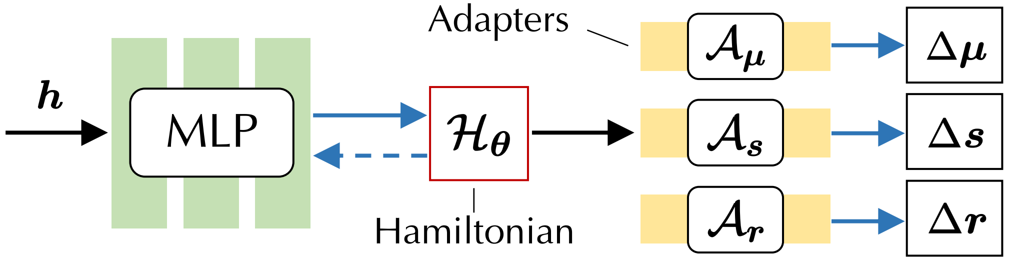



**Figure 1.** Hamiltonian Neural Network Decoder

$$\boldsymbol{v}_c = \nabla_{\boldsymbol{h}} F_1(\boldsymbol{h}) \boldsymbol{I}, \quad \boldsymbol{v}_s = \nabla_{\boldsymbol{h}} F_2(\boldsymbol{h}) \boldsymbol{M}^\top$$

where $\boldsymbol{v}_c$ preserves energy and $\boldsymbol{v}_s$ preserves volume.

- **Linear Adaptors Reshape Output Vector Fields:** Lightweight attribute-specific linear layers process HNN-generated vector fields, maintaining standard dimensionality of each Gaussian attribute.

$$\Delta \boldsymbol{\mu} = \mathcal{A}_{\mu} \boldsymbol{v}, \quad \Delta \boldsymbol{s} = \mathcal{A}_{s} \boldsymbol{v}, \quad \Delta \boldsymbol{r} = \mathcal{A}_{r} \boldsymbol{v}$$

where $\Delta \boldsymbol{\mu}$, $\Delta \boldsymbol{s}$, and $\Delta \boldsymbol{r}$ are position, scaling, and rotation.

[1] Greydanus S, Dzamba M, Yosinski J. Hamiltonian Neural Networks. NeurIPS 2019.

# Boltzmann Equilibrium Decomposition

**Why Decomposition?** Deforming all primitives is inefficient; only non-equilibrium Gaussians should deform.

**Our Implementation:** Constructing soft masks based on Boltzmann energy distributions that adaptively filter out primitives useless to deformation.

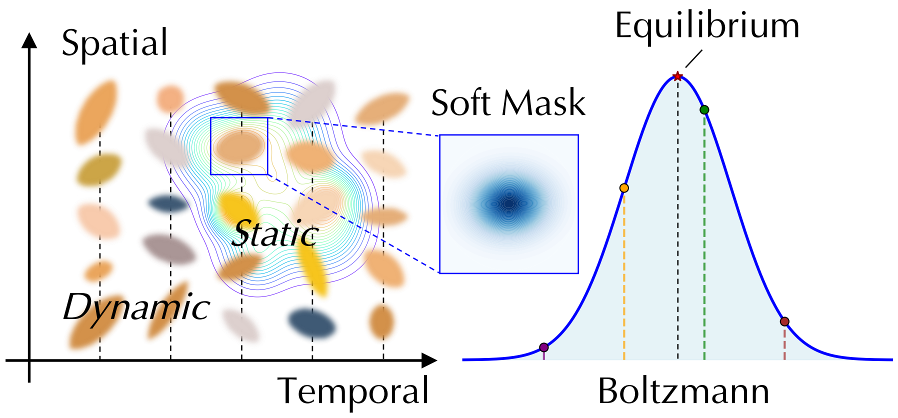



**Figure 2.** Boltzmann Equilibrium Decomposition

1. **Position Dynamics (Spatial-Temporal)**
   - **Harmonic Oscillator Model:** Modeling energy deviation $E_{st}$
   - **Mask:** $M_{pos} = (1-\gamma)\cdot\exp(-\beta E_{st}) + \gamma$
2. **Scaling Dynamics (Temporal-Only)**
   - **Why Temporal-Only:** Scaling mainly affects surface detail, not global structure. Modeling energy deviation $E_t$
   - **Mask:** $M_{scale} = (1-\gamma)\cdot\exp(-\beta E_t) + \gamma$

- **Attribute-Specific Blending with Soft Masks:** Applying decomposition strategies based on attributes.

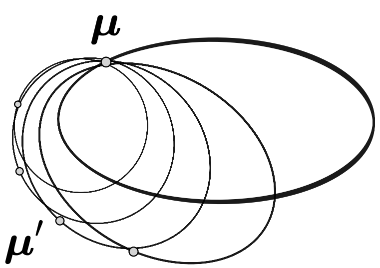
$\boldsymbol{\mu}' = \boldsymbol{\mu} + \Delta\boldsymbol{\mu} \odot (1 - M_{pos}), \quad \boldsymbol{s}' = \boldsymbol{s} + \Delta\boldsymbol{s} \odot (1 - M_{scale})$ where $\odot$ represents the Hadamard product.

# Physics-Informed Constraints

**Why Constraints?** Real-world dissipation breaks energy conservation, demanding physics-informed constraints.

**Our Implementation:** Second-order symplectic integration for position dynamics and local rigidity regularization for rotation dynamics.

### 1. Position Dynamics (Symplectic Integration)



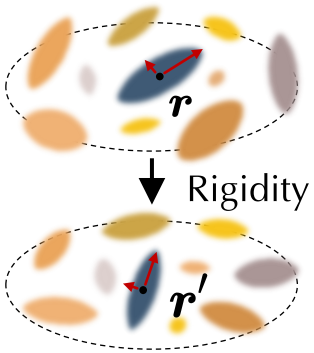

Symplecticity

- **Problem:** Euler causes energy drift.
- **Solution:** Position Verlet scheme.

$$\tilde{\boldsymbol{\mu}} = \boldsymbol{\mu} + \Delta t \cdot \Delta\boldsymbol{\mu} + \frac{(\Delta t)^2}{2}\boldsymbol{F}$$

where force $\boldsymbol{F}$ is from conservative $\boldsymbol{v}_c$ .

### 2. Rotation Dynamics (Rigidity Regularization)




- **Inspiration:** As-Rigid-As-Possible.
- **Approach:** Clamp Gaussian rotation magnitude utilizing tanh to prevent unnatural twisting.

$$\phi' = \phi_{max} \cdot \tanh\left(\frac{\phi}{\phi_{max}}\right)$$

- **Note:** Applying physics-informed constraints before Boltzmann equilibrium decomposition.

$$\boldsymbol{\mu}' = \tilde{\boldsymbol{\mu}} \odot (1 - M_{pos}) + \boldsymbol{\mu} \odot M_{pos}$$

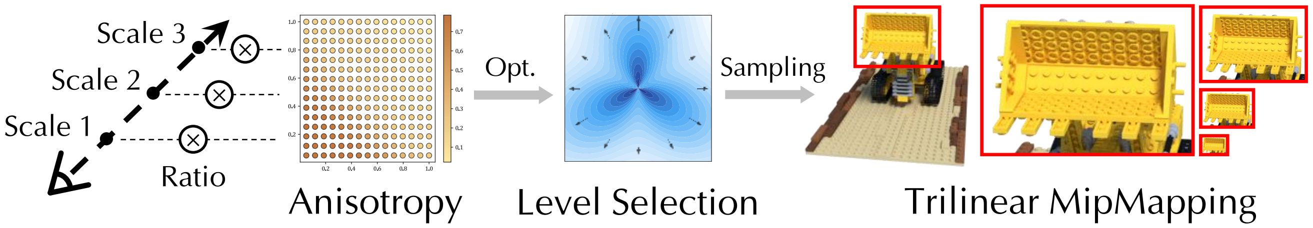
where $\tilde{\boldsymbol{\mu}}$ is the position regularized by symplectic integration.


[2] Sorkine O, Alexa M. As-Rigid-As-Possible Surface Modeling. Symposium on Geometry Processing 2007.

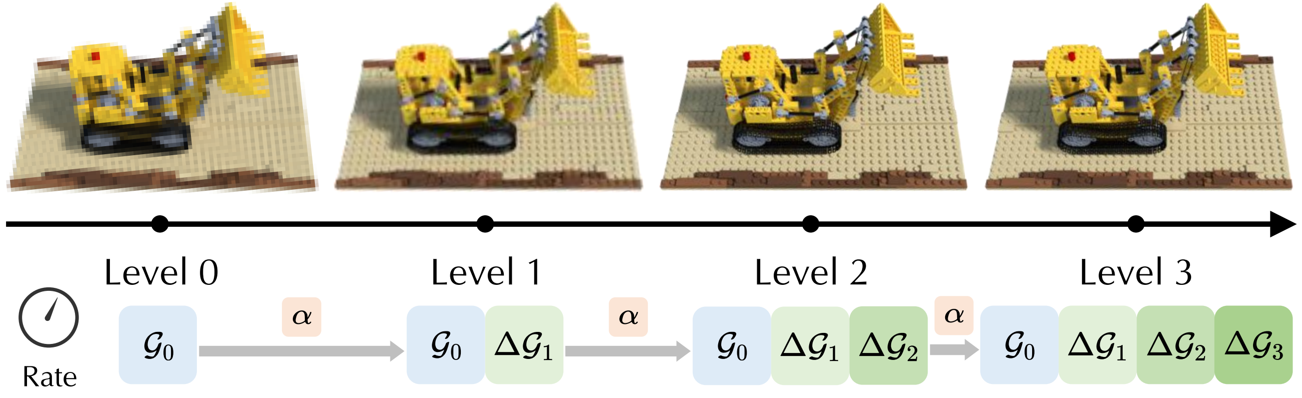
# Adapting NeHaD to Adaptive Streaming

**Application:** Bandwidth-constrained adaptive VR streaming.

**Our Implementation:** Scale-aware anisotropic MipMapping and layered progressive optimization for multi-level LOD rendering.




(a) The pipeline of scale-aware anisotropic MipMapping for anti-aliasing




(b) The pipeline of layered progressive optimization for adaptive streaming

**Figure 3.** Adapting NeHaD to Adaptive Streaming

## 1. Scale-Aware Anisotropic MipMapping

- **Approach:** Scale analysis via anisotropic weights guides mipmap-based trilinear sampling.

$$\hat{\boldsymbol{l}} = \boldsymbol{L} - \beta(\rho) \cdot (\boldsymbol{L} - \bar{L}\mathbf{1}), \quad \beta(\rho) = \frac{\tanh(\rho/3 - 1)}{1 + \tanh(\rho/3 - 1)}$$

## 2. Layered Progressive LOD Optimization

- **Approach:** Coarse-to-fine optimization, progressively training Gaussian splats from lowest to highest resolution across various levels.

$$\mathcal{G}_i = \mathcal{G}_0 + \sum_{j=1}^{i} \Delta\mathcal{G}_j, \quad j \in \{1, 2, \ldots, N\}$$

# Experimental Results

**Remark:** NeHaD outperforms state-of-the-art methods across all evaluation metrics while maintaining over 20 FPS, balancing visual quality and rendering efficiency.

**Table 1.** Quantitative Results

| D-NeRF [Pumarola et al. 2021] (monocular, synthetic, 800×800) | | | | | |
|---|---|---|---|---|---|
| Method | PSNR ↑ | SSIM ↑ | LPIPS ↓ | Train Time ↓ | FPS ↑ |
| D-NeRF | 29.68 | 0.947 | 0.058 | 48*hrs* | <1 |
| TiNeuVox | 32.74 | 0.972 | 0.051 | 28*min* | 1.5 |
| K-Planes | 31.52 | 0.967 | 0.047 | 52*min* | 0.97 |
| HexPlane | 31.04 | 0.97 | 0.04 | **11.5*min*** | 2.5 |
| 4DGS | 35.34 | 0.985 | 0.021 | 20*min* | 82 |
| SC-GS | 40.26 | 0.992 | 0.009 | 30*min* | **164** |
| Ours | **40.91** | **0.995** | **0.008** | 24*min* | 62 |
| **HyperNeRF [Park et al. 2021b] (monocular, real-world, 536×960)** | | | | | |
| Method | PSNR ↑ | MS-SSIM ↑ | LPIPS ↓ | Train Time ↓ | FPS ↑ |
| HyperNeRF | 22.41 | 0.814 | 0.131 | 32*hrs* | <1 |
| TiNeuVox | 24.20 | 0.836 | 0.128 | **30*min*** | 1 |
| 4DGS | 25.24 | 0.845 | 0.116 | 34*min* | 32 |
| DeformGS | 25.02 | 0.822 | 0.116 | 1.5*hrs* | 13 |
| SaRO-GS | 25.38 | 0.850 | 0.110 | 1.2*hrs* | 34 |
| Grid4D | 25.50 | 0.856 | 0.107 | 2.5*hrs* | **37** |
| Ours | **25.69** | **0.858** | **0.104** | 45*min* | 25 |
| **DyNeRF [Li et al. 2022] (multi-view, real-world, 1352×1014)** | | | | | |
| Method | PSNR ↑ | D-SSIM ↓ | LPIPS ↓ | Train Time ↓ | FPS ↑ |
| DyNeRF | 29.58 | 0.020 | 0.083 | 1344*hrs* | <1 |
| HexPlane | 31.70 | 0.014 | 0.075 | 12*hrs* | 0.2 |
| 4DGS | 31.17 | 0.016 | 0.049 | 42*min* | 30 |
| STG | 32.05 | 0.014 | 0.044 | 10*hrs* | 110 |
| SaRO-GS | 32.15 | 0.014 | 0.044 | 1.5*hrs* | 32 |
| Swift4D | 32.23 | 0.014 | 0.043 | **25*min*** | **125** |
| Ours | **32.35** | **0.013** | **0.042** | 50*min* | 21 |

**Table 2.** Quantitative Ablation Study

| | Model configuration | PSNR ↑ | SSIM ↑ | LPIPS ↓ |
|---|---|---|---|---|
| | Baseline model | 35.34 | 0.985 | 0.021 |
| + | Hamiltonian neural network | 37.69 | 0.989 | 0.012 |
| + BED | Spatial-temporal decomposition | 37.46 | 0.989 | 0.014 |
| + BED | Temporal-only decomposition | 36.08 | 0.986 | 0.019 |
| + PIC | Symplectic integration | 36.53 | 0.986 | 0.018 |
| + PIC | Local rigidity regularization | 36.04 | 0.986 | 0.020 |
| | Proposed model | 40.91 | 0.995 | 0.008 |

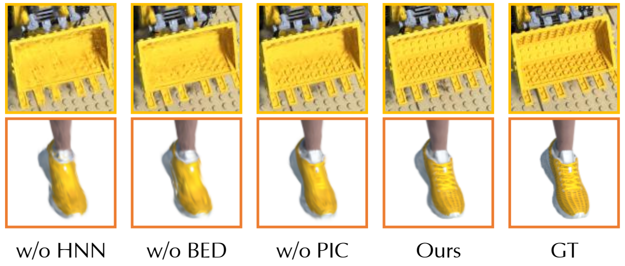



**Figure 4.** Qualitative Ablation Study

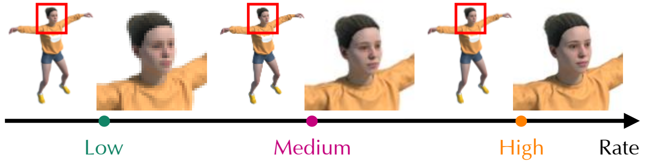



**Figure 5.** Streaming Results

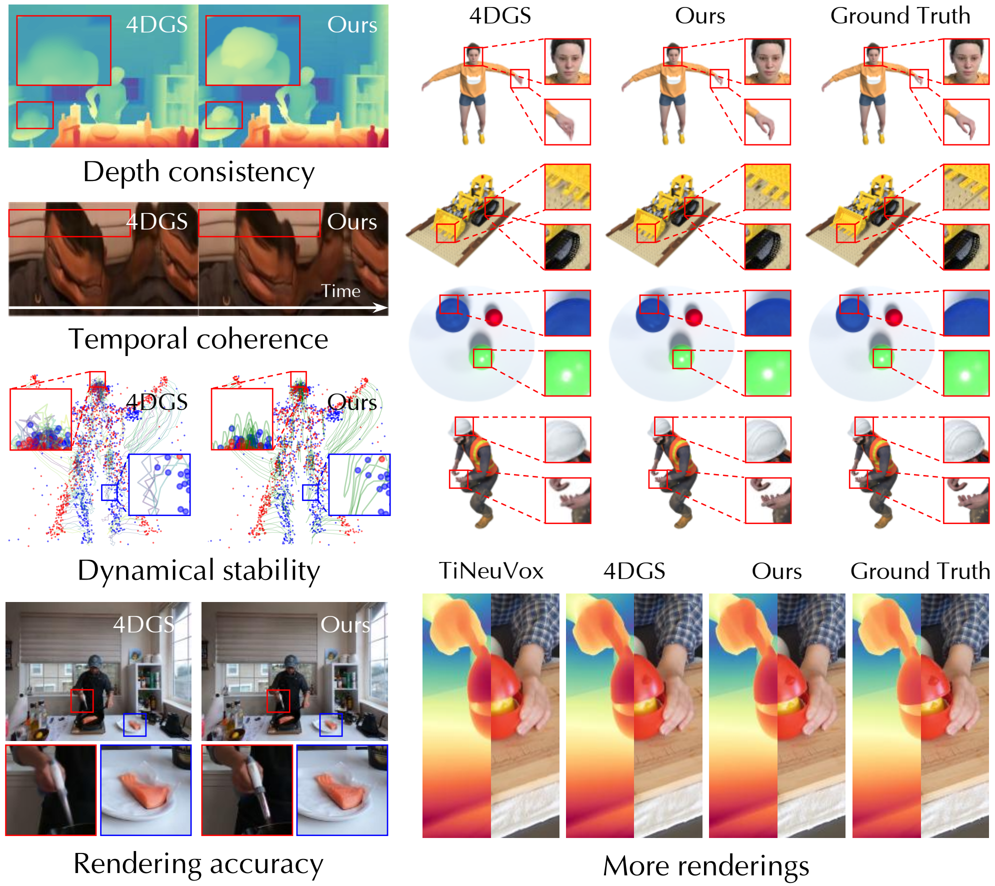



**Figure 6.** Qualitative Results

# Conclusion

## Summary

1. **Problem** Existing methods struggle with complex dynamics and physically implausible motions.
2. **Idea** Hamiltonian mechanics naturally fits Gaussian deformation via symplectic manifold structure.
3. **Method (3 Innovative Components Improving the Baseline 4DGS)**
   - Hamiltonian neural networks implicitly learn conservation laws, ensuring stable deformations without sudden discontinuities.
   - Adaptively separates static and dynamic Gaussians based on spatial-temporal energy states.
   - Second-order symplectic integration and local rigidity constraints handle real-world dissipative forces.
4. **Contribution** First Hamiltonian-based Gaussian deformation field with physically plausible rendering and streaming capability.

## Limitation & Future Work

- Computational overhead from Hamiltonian derivatives.
- Complex non-conservative forces (*e.g.*, fluid dynamics) remain challenging.

[3] Wu G, et al. 4D Gaussian Splatting for Real-Time Dynamic Scene Rendering. CVPR 2024.

SIGGRAPH ASIA 2025 HONG KONG 香港

GENERATIVE RENAISSANCE

# Thank You!