PhysGaussian: Physics-Integrated 3D Gaussians for Generative Dynamics

Revisor
Mohara Nascimento
Instituto Tecgraf/PUC-Rio
mohara.civil@gmail.com

Archaeologist
Leonardo Mendonça
IMPA
leonardo.mendonca@impa.br

Hacker
Marcelo de Sousa
Universidade de São Paulo
marcelo.sousa@usp.br

PhD Student
Marcelo de Sousa
Universidade de São Paulo
marcelo.sousa@usp.br

1. Revisor

PhysGaussian is a method that aims to integrate a physics-based dynamic model within a Gaussian Splatting representation, following the paradigm "what you see is what you simulate (WS²)". With this intention, the paper combines the 3D Gaussian Splatting reconstruction along with Gaussians evolution and Continuum Mechanics principles. This latter is embedding with a mechanical method called Material Point Method (MPM).

1.1. Overview

The method begins creating a 3D Gaussian Splatting reconstruction with the 3DGS framework [4], receiving as input images and camera information. At this stage, PhysGaussian introduces an anisotropic loss term, intending to avoid over-skinny Gaussians kernels. With the reconstructed scene the method treats the Gaussian Ellipsoids as a Continuum and embedding physical characteristics (mass, velocity, etc.) and material properties (elasticity modulus, poisson's ratio, constitutive model, etc.) in each Gaussian kernel. To turn the Gaussians time dependents the Gaussians are treated as particles and their original kernels are deformed according with a deformation gradient tensor (F), as shown in the Equation 1.

$$G_p(x,t) = e^{-\frac{1}{2}(\mathbf{x}-\mathbf{x_p})^T (F_p A_p F_p^T)^{-1}(\mathbf{x}-\mathbf{x_p})} \tag{1}$$

where $G_p(x,t)$ is the Gaussian deformed kernel in a instant t, x_p and A_p are, respectively, the center and the covariance matrix, both time-dependent, and F_p is the particle deformation gradient tensor.

Beyond transform Gaussian in particles, some other treatments are applied to produce a physical appearance, as rotate the orientations of spherical harmonics when the world-space is rotated and internal filling the Gaussians. After all these considerations, the motion scene is generated. The Figure 1 summarize this with an overview of the method.

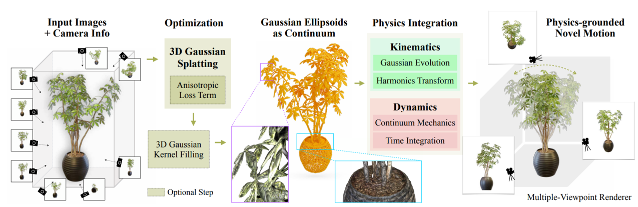

Figure 1. PhysGaussian method overview.

1.2. Strengths

- Physical Method choice: the choice of MPM possibilities the use of the 3DGS framework integrated to the physics behavior, in a natural form;
- Concerns about choosing the adequate constitutive model according to the material;
- Despite to propose a new and innovative approach, the paper uses two consolidated methods in their fields (MPM and 3DGS);
- The method captures volumetric behaviors.

1.3. Weaknesses

- Some demos present unrealistic material behavior, with problems related to the form as the movement affects the structure components;

- The contact between materials must be better evaluated, considering the influence of one region in another;
- Blur in fractured regions.

1.4. Evaluation

The rating attributed to the paper is 5, the work is concise, clear and achieve what is proposed, also the method is innovative and efficient. Moreover, the authors were concerned in represent real material in the more realistic form, inputting adopted constitutive models and properties in different scenarios and materials.

2. Archaeologist

2.1. Previous works

PhysGaussian [8] has the purpose of modeling dynamic scenes from real images of a given static scene. In order to do so, it combines methods from 3DGS [4], which models a scene through 3D gaussians, whose color, position and scales are optimized through gradient descent, along with the Material-Point-Method [6], a continuum mechanics technique for physical simulation, which has also seen success in animation [5]. Figs. 2 and 3 show the timeline of developments in these two areas, leading to PhysGaussian.

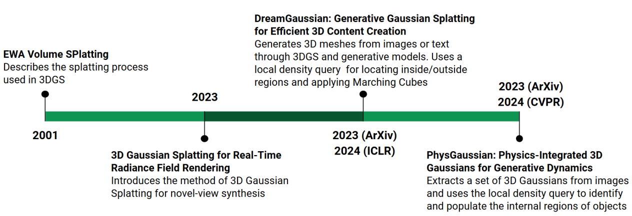

Figure 2. Timeline of Gaussian Splatting-related papers that contribute to the development of PhysGaussian

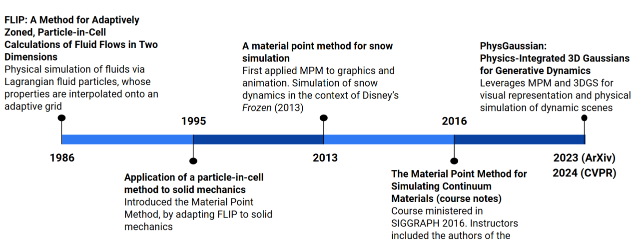

Figure 3. Timeline of works in the Material Point Method that are relevant to PhysGaussian

FLIP: A Method for Adaptively Zoned, Particle-in-Cell Calculations of Fluid Flows in Two Dimensions[1]. This methodology, introduced in 1986, models a fluid as a set of Lagrangian particles, each with a respective position, velocity, mass and energy. At each timestep, before any calculations can be done, the particle properties are projected onto a quadrilateral grid. Then, the motion equations are solved numerically on the grid, and subsequently re-projected back to the particle cloud. This simulation methodology was originally designed for numerically solving fluid dynamics problems.

MPM: Application of a particle-in-cell method to solid mechanics[6]. This paper (1995) extends FLIP to the simulation of solid materials. The particle discretization allows for natural modeling of discontinuities, such as fractures or interfaces, as well as large deformations and history-dependent material behavior. These physical scenarios are of interest to engineering problems, but are difficult to solve using the Finite Element Method for continuum mechanics.

A material point method for snow simulation[5]. The first application of the Material Point Method to computer graphics[1]. It was developed during the making of Disney's *Frozen*, and seeks to animate snow in a realistic way that's coherent with solid mechanics. MPM is used to model the motion of snow as an elasto-plastic flow, using a constitutive equation for snow derived from engineering models, and calibrated for visually interesting and realistic/verisimilar scenes. Some modifications are also made to allow for better animator control over snow's visual behavior.

DreamGaussian: Generative Gaussian Splatting for Efficient 3D Content Creation[7]. The only Gaussian Splatting paper, other than [4], used by PhysGaussian. It employs generative models to create a set of views from a single image or a text prompt, and subsequently optimizes a set of 3D gaussians for these views. Finally, a mesh is extracted through Marching Cubes and a local density query:

$$d(\mathbf{x}) = \sum_i \alpha_i e^{-\frac{1}{2}(\mathbf{x}-\mathbf{x_i})^T \Sigma_i^{-1}(\mathbf{x}-\mathbf{x_i})} \tag{2}$$

This density query is the only part of DreamGaussian that's incorporated into PhysGaussian.

2.2. Current works

VR-GS: A physical dynamics-aware interactive gaussian splatting system in virtual reality[3]. VR-GS seeks to allow real-time interactive scenes with physics-based deformation of solids, for virtual reality applications. Unlike

[1]Some of this paper's authors also went on to minister a course on MPM for animation during SIGGRAPH 2016[2], spreading the method among the computer graphics community.

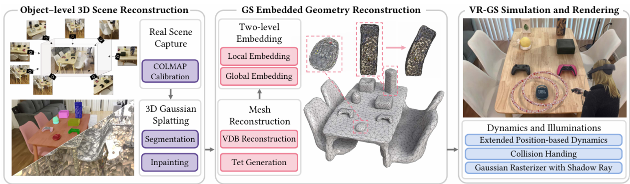

Figure 4. Overview of the pipeline for VR-GS. Source: [3]

PhysGaussian, VR-GS embeds each Gaussian into a tetrahedral mesh, and then simulates the deformation through XPBD, a preexisting physics simulator. An overview of the method is shown in fig. 4. VR-GS uses PhysGaussian as a benchmark, and seeks to achieve similar visual results at a lower computational cost. The authors claim a success in doing so, however no quantitative experiment was shown making this comparison.

3. Hacker: Summary of the PhysGaussian Pipeline and Simulations

The PhysGaussian pipeline is a comprehensive framework designed to integrate Gaussian Splatting and the Material Point Method (MPM) for creating high-fidelity physical and visual simulations. This pipeline supports simulations across a wide range of applications, enabling the study of complex material behaviors and dynamic interactions. Its stages ensure the precise execution of both physical and graphical computations, structured as follows:

1. **Configuration Loading:** At this stage, the system imports detailed scene configurations and model parameters. These configurations include boundary definitions, initial conditions, and material properties required to define the simulation environment accurately.
2. **Solver Initialization:** The MPM solver is configured to handle advanced physical computations, such as particle motion, material deformation, and collision responses. This solver ensures that simulations are physically accurate and computationally efficient.
3. **Boundary Conditions Definition:** Environmental constraints are applied to define interactions between simulated particles and their surrounding scene. This includes interactions with solid objects, external forces, and environmental effects such as gravity.
4. **Rendering and Visualization:** The final stage converts simulation results into visual outputs, generating detailed images or videos that accurately reflect the dynamic processes and deformations observed during the simulation.

The pipeline leverages advanced libraries, such as Warp, which significantly accelerate GPU computations. This enables the efficient handling of large datasets and complex simulations. Several Python modules are integral to this process:

- `mpm_solver_warp.py`: Handles the core physical computations, including particle-to-grid (P2G) transfers and deformation updates.
- `diff_gaussian_rasterization.py`: Manages the rendering of Gaussian particles, ensuring that dynamic changes are visually accurate.
- `internal_filling.py`: Focuses on filling volumetric regions of objects, ensuring uniformity and correcting inconsistencies in simulations.

The implementation includes sophisticated processes such as collision detection using ray-based methods and dynamic updates of particle properties. These updates ensure continuous refinement of parameters like covariance matrices, deformation gradients, and applied forces. The system also incorporates solutions to challenging problems, such as filling hidden regions within volumetric objects and addressing computational inconsistencies in complex geometries. This includes ensuring volumetric continuity in scenarios with irregular shapes or materials, a critical feature for applications such as soft-body dynamics or fluid simulations.

In addition, the pipeline includes tools for diagnosing and correcting simulation errors. For example, debug modes highlight discrepancies in physical parameters or inconsistencies in boundary conditions, streamlining the resolution of simulation challenges. Integration with visualization tools also aids researchers in evaluating simulation results in real-time, enabling iterative improvements to scene parameters.

Despite its robustness, the pipeline encounters challenges, including the complexity of JSON configurations, errors linked to CUDA and Taichi, and limited documentation for some physical parameters. The reliance on highly specific hardware, such as GPUs optimized for parallel processing, may also limit accessibility. Nonetheless, the PhysGaussian pipeline excels in delivering high-quality simulations. It allows researchers to explore dynamic behaviors in materials under diverse scenarios, pushing the boundaries of computational physics and visualization.

4. PhD Student: Summary of the Dynamic Mesh Framework

The dynamic mesh adaptation framework is an advanced system developed to address challenges in the 3D representation of dynamic objects, particularly under extreme deformations. By treating Gaussian centers as vertices within a dynamic triangular mesh, the framework ensures precise control over deformations while maintaining computational efficiency. The methodology is extensively described in the paper *A Simple and Flexible Framework to Adapt Dynamic Meshes*.

The primary issues the framework resolves include:

- **Excessive Anisotropy:** In scenarios of high deformation, Gaussian kernels can become excessively elongated, leading to visual distortions and inaccurate representations.
- **Inadequate Distribution:** Uneven scaling along Gaussian axes causes errors in geometry and physics simulations, compromising overall simulation fidelity.

To mitigate these problems, the framework incorporates dynamic structural operations:

1. **Refinement (Edge Split):** Edges in highly deformed or geometrically complex regions are split to increase resolution and accuracy.
2. **Simplification (Vertex Weld):** Stable areas with minimal deformation are simplified by merging vertices, reducing computational overhead while maintaining accuracy.
3. **Curvature-Sensitive Laplacian Smoothing:** This process adjusts vertex positions based on local curvature, preserving critical details in regions with high geometric complexity.

Dynamic mesh adaptation occurs continuously throughout the simulation process, with real-time adjustments made to ensure fidelity and numerical stability. At each time step, vertex properties, including positions, connectivity, and Gaussian attributes, are updated based on deformation gradients and curvature thresholds. These updates enable the framework to handle complex scenarios such as tearing, folding, or re-meshing in highly dynamic environments.

The framework's flexibility extends to applications in real-time rendering, where it balances computational efficiency with visual quality. For example, regions of high deformation density are allocated additional computational resources, while stable areas are simplified to conserve memory and processing power. This selective adaptation enhances the overall efficiency of simulations, making them suitable for both academic research and practical applications, such as real-time VR simulations or engineering analysis.

The benefits of the framework are multifaceted:

- **Visual Enhancement:** Reduces artifacts and improves detail preservation in critical areas.
- **Computational Efficiency:** Simplified regions consume fewer resources, enabling faster simulations without compromising accuracy.
- **Numerical Stability:** Prevents issues such as poorly conditioned matrices during calculations, ensuring consistent results even in high-stress simulations.
- **Real-Time Adaptation:** Continuously adjusts mesh structures to accommodate dynamic deformations and maintain consistency.

Future enhancements to the framework include exploring support for complex topological changes, such as dynamic creation or removal of mesh elements, to better model scenarios involving fracture or merging. Another direction involves automating parameter selection, enabling users to define broad criteria, with the framework dynamically tuning thresholds for curvature, deformation, and connectivity. These improvements aim to extend the adaptability and user-friendliness of the dynamic mesh framework, positioning it as an essential tool for simulations in PhysGaussian and beyond.

5. Conclusion

In summary, the proposal of the PhysGaussian is to recreate a 3D scene at the same time that consider physics characteristics and motion. To achieve this, the paper introduces an innovative approach of transform Gaussian kernel in particles and inserting a mechanical method into it. The authors were successful in their objective, PhysGaussian is capable to represent satisfactorily movements of different scenes through images and physical information. As future works is important to focus on the material boundary conditions, improving it and creating a more realistic result.

References

[1] Jeremiah U Brackbill and Hans M Ruppel. Flip: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions. *Journal of Computational physics*, 65(2): 314–343, 1986. 2

[2] Chenfanfu Jiang, Craig Schroeder, Joseph Teran, Alexey Stomakhin, and Andrew Selle. The material point method

for simulating continuum materials. In *Acm siggraph 2016 courses*, pages 1–52. 2016. 2

[3] Ying Jiang, Chang Yu, Tianyi Xie, Xuan Li, Yutao Feng, Huamin Wang, Minchen Li, Henry Lau, Feng Gao, Yin Yang, et al. Vr-gs: A physical dynamics-aware interactive gaussian splatting system in virtual reality. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–1, 2024. 2, 3

[4] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 1, 2

[5] Alexey Stomakhin, Craig Schroeder, Lawrence Chai, Joseph Teran, and Andrew Selle. A material point method for snow simulation. *ACM Transactions on Graphics (TOG)*, 32(4):1–10, 2013. 2

[6] Deborah Sulsky, Shi-Jian Zhou, and Howard L Schreyer. Application of a particle-in-cell method to solid mechanics. *Computer physics communications*, 87(1-2):236–252, 1995. 2

[7] Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. *arXiv preprint arXiv:2309.16653*, 2023. 2

[8] Tianyi Xie, Zeshun Zong, Yuxing Qiu, Xuan Li, Yutao Feng, Yin Yang, and Chenfanfu Jiang. Physgaussian: Physics-integrated 3d gaussians for generative dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4389–4398, 2024. 2