

# Gaussian Swaying 🌿: Surface-Based Framework for Aerodynamic Simulation with 3D Gaussians

## Supplementary Material

### 1. MPM Algorithm

MPM simulates dynamic movements via discretizing a continuous body into particles and then updating their properties (including positions, velocities, etc.). We summarize the MPM algorithm as follows:

1. **Transfer Particles to Grid.** The mass and momentum from particles are transformed to grid nodes as

$$\begin{aligned} m_i^n &= \sum_p w_{ip}^n m_p, \\ m_i^n \mathbf{v}_i^n &= \sum_p w_{ip}^n m_p (\mathbf{v}_p^n + \mathbf{C}_p^n (\mathbf{x}_i - \mathbf{x}_p^n)), \end{aligned} \quad (1)$$

where  $\mathbf{C}_p^n$  refers to affine momentum [1] on particle  $p$ .

2. **Grid Update.** Update grid velocities based on forces at the next timestep by

$$\frac{m_i}{\Delta t} (\mathbf{v}_i^{n+1} - \mathbf{v}_i^n) = - \sum_p \tau_p^n \nabla w_{ip}^n V_p^0 + \mathbf{f}_i^{ext}, \quad (2)$$

3. **Transfer Grid to Particles.** Transfer velocities back to particles and update particle states.

$$\begin{aligned} \mathbf{v}_p^{n+1} &= \sum_i \mathbf{v}_i^{n+1} w_{ip}^n, \\ \mathbf{x}_p^{n+1} &= \mathbf{x}_p^n + \Delta t \mathbf{v}_p^{n+1}, \\ \mathbf{C}_p^{n+1} &= \frac{12}{\Delta x^2 (b+1)} \sum_i w_{ip}^n \mathbf{v}_i^{n+1} (\mathbf{x}_i^n - \mathbf{x}_p^n)^T, \\ \nabla \mathbf{v}_p^{n+1} &= \sum_i \mathbf{v}_i^{n+1} \nabla w_{ip}^n, \\ \mathbf{F}_p^{E,n+1} &= \mathcal{Z}((\mathbf{I} + \nabla \mathbf{v}_p^{n+1}) \mathbf{F}_p^{E,n}), \\ \tau_p^{n+1} &= \tau(\mathbf{F}_p^{E,n+1}). \end{aligned} \quad (3)$$

Here  $b$  is the B-spline degree, and  $\Delta x$  is the Eulerian grid spacing. The deformation gradient  $\mathbf{F}_p = \mathbf{F}_p^E \mathbf{F}_p^P$  is decomposed into elastic part  $\mathbf{F}_p^E$  and plastic part  $\mathbf{F}_p^P$ . Detailed computation of the return map  $\mathcal{Z}$  and the Kirchhoff stress  $\tau$  is listed in Sec. 2. Please refer to [2, 6] for more details.

### 2. Details on Elastic and Plastic Models

In this section, we elaborate on the elastic and plastic constitutive models used in Gaussian Swaying, as listed in Tab. 1. We adopt the models from [6].

#### 2.1. Fixed Corotated Elasticity

The Kirchhoff stress  $\tau$  is defined as

$$\tau_p = 2\mu(\mathbf{F}_p^E - \mathbf{R}_p) \mathbf{F}_p^{E^T} + \lambda(J_p - 1)J_p, \quad (4)$$

where  $\mathbf{R}_p = \mathbf{U}_p \mathbf{V}_p^T$  denotes local rotation and  $\mathbf{F}_p^E = \mathbf{U}_p \Sigma_p \mathbf{V}_p^T$  is the singular value decomposition of elastic deformation gradient.  $J_p$  is the determinant of  $\mathbf{F}_p^E$  [1].

#### 2.2. Drucker-Prager Plasticity

Drucker-Prager plasticity is used for simulation of sand [4].

We set friction angle as  $\phi_f$  and  $\alpha = \sqrt{\frac{2}{3}} \frac{2 \sin \phi_f}{3 - \sin \phi_f}$ . We set  $\phi_f = 30^\circ$  in the experiment. Then we calculate intermediary variants:

$$\hat{\epsilon}_p = \epsilon_p - \frac{\text{tr}(\epsilon_p)}{d} \mathbf{I}, \quad \delta\gamma = \|\hat{\epsilon}_p\|_F + \alpha \frac{(d\lambda + 2\mu)\text{tr}(\epsilon_p)}{2\mu}, \quad (5)$$

where  $\mathbf{F}_p^E = \mathbf{U}_p \Sigma_p \mathbf{V}_p^T$  and  $\epsilon_p = \log(\Sigma_p)$ .  $d$  is the dimension and  $\hat{\epsilon}_p$  is the plastic deformation amount.

The detailed calculation of return mapping  $\mathcal{Z}$  is given as

$$\mathbf{F}_p^E = \mathbf{U}_p \mathcal{Z}(\Sigma_p) \mathbf{V}_p^T, \quad (6)$$

Here, if  $\text{tr}(\epsilon_p) > 0$ , then  $\mathcal{Z}(\Sigma_p) = 1$ .  $\mathcal{Z}(\Sigma_p) = \Sigma_p$  if  $\hat{\epsilon}_p \leq 0$  and  $\text{tr}(\epsilon_p) \leq 0$ . For other conditions,  $\mathcal{Z}$  is calculated as  $\exp\left(\epsilon_p - \delta\gamma_p \frac{\epsilon_p}{\|\epsilon_p\|}\right)$ .

### 3. More Details on Parameters

We further provide detailed parameter settings, including Young's modulus  $E$ , Poisson's ratio  $\nu$ , mass density  $\rho$ , aerodynamic force coefficients  $\mathbf{C}_D$ ,  $\mathbf{C}_F$ ,  $\mathbf{C}_L$ , flow intensity and constitutive model of material in Tab. 1. The constitutive model is defined the same with [5]. If the flow intensity changes during the simulation, it will be denoted by two values in the table (*Cloth* and *Vase* in Tab. 1).

By default, we add three-dimension Gaussian noise with mean value 0 and standard deviation 0.3 to the wind speed to best approximate the behavior of the natural wind. The biggest component of the wind speed (e.g. 1.7 in *Leaves* in Tab. 1) also serves the mean value of a sine function. The amplitude of the sine function is set to half of this component's value to simulate the swaying effect. In the flag simulation dataset, gravity is set as  $9.8m/s^2$ .

We train 3D Gaussians using the official implementation of [3] with proposed training losses. In Tab. 2, we showcase

Table 1. **Parameter Settings.** We list the parameter settings on scenes in Gaussian Swaying.

Scene	Material				Aerodynamics			
	Constitutive Model	$E$	$\nu$	$\rho$	$C_D$	$C_F$	$C_L$	Flow Intensity
<i>Ficus</i> (Truck)	Fixed Corotated	2e6	0.4	300	0.5	0.4	0.01	(1.7, 0, 0)
<i>Leaves</i>	Fixed Corotated	1e4	0.4	15	0.5	0.4	0.01	(1.7, 0, 0)
<i>Sand</i>	Drucker-Prager	5e5	0.3	200	0.4	0.3	0.01	(10, 0, 0)
<i>Telephone</i>	Fixed Corotated	5e5	0.4	200	0.5	0.4	0.01	(-1, 0, 1)
<i>Alocasia</i>	Fixed Corotated	2e6	0.4	300	0.5	0.4	0.01	(0, -0.2, 0)
<i>Vase</i>	Fixed Corotated	1e4	0.3	20	0.4	0.3	0.005	(0, 0, -1.5), (0, 0, 1.5)
<i>Flag</i> (Pattern-1)	Fixed Corotated	5e3	0.3	50	-	-	-	(0, 0, 0)
<i>Flag</i> (Pattern-2)	Fixed Corotated	3e3	0.3	30	0.1	0.3	0.005	(2.5, 0.5, 0)
<i>Flag</i> (Pattern-3)	Fixed Corotated	3e3	0.3	20	0.1	0.3	0.005	(2, 0, 0)

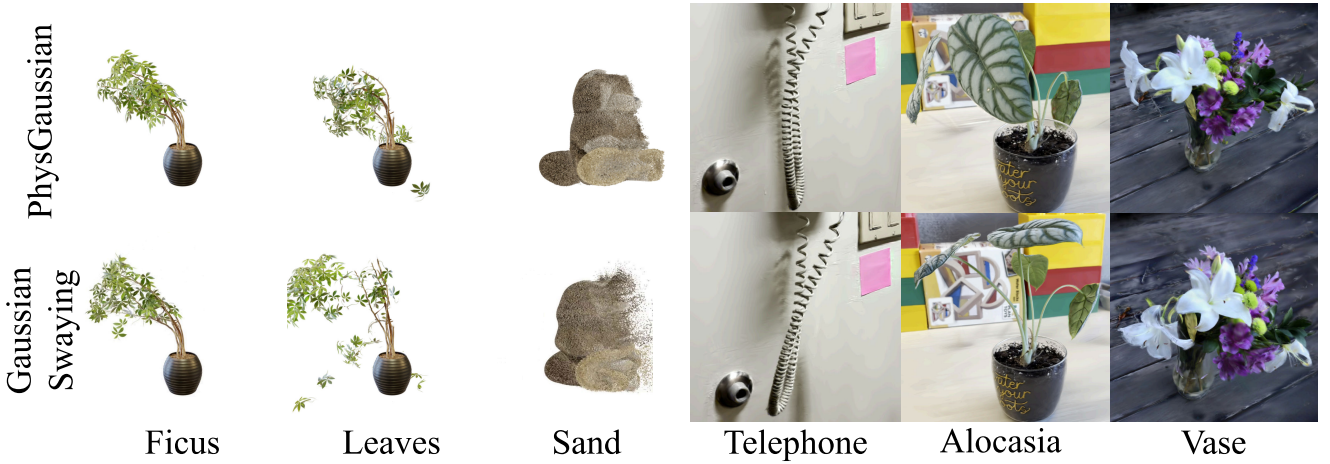


Figure 1. Qualitative comparisons between PhysGaussian (*Upper Row*) and Gaussian Swaying (*Lower Row*).

ablation studies on Gaussian size, where  $b = 0.008$  is optimal for our setting. The total iterations are 60,000. In the simulation, Gaussians are filtered by an opacity threshold of 0.1 (In *Sand* we set it as 0.02). The frame number is 250, with each frame generated every 0.04s. The frame rate for the video is 25 for all experiments.

Table 2. **Ablation studies on Gaussian size.**

Param $b$	0.001	0.005	0.008	0.01	0.05	0.1
FVD ↓	362.77	248.66	<b>219.94</b>	432.88	796.01	910.24

## 4. More Results

We visualize qualitative comparisons with PhysGaussian in Fig. 1. As shown, PhysGaussian fails to capture surface-level interaction (*e.g.*, unrealistic leaf orientations in *Ficus* and *Leaves*, sand stretching rightward without scattering in *Sand*).

*Vase*, *Telephone* and *Alocasia* are real-world datasets to

validate the application of Gaussian Swaying. In *Telephone* and *Alocasia*, objects are applied with a gentle leftward incident flow. For comparisons, PhysGaussian unnaturally overstretches alocasia and telephone line, while ours accurately models leaf orientation and telephone line floating in the wind.

## 5. Object Editing

To further demonstrate the flexibility and versatility of Gaussian Swaying, we explore its capability for real-time scene editing, including adjustments to material properties, colors, and interaction parameters. In Fig. 2, the *Upper Row* is *Sand* while the *Lower Row* is edited to *Foam* with white color. Foam generally is stickier than sand, which aligns with the generated results by Gaussian Swaying.

## 6. Video Results

We provide video results in the supplementary materials. For example, detailed surface-level interactions between objects and wind can be best reflected in ‘Ficus.mp4’ and



Figure 2. **Object Editing.** Gaussian Swaying is capable of realistic material and color editing to facilitate versatile applications. The top sand bear is edited to white foam at the bottom.

‘Leaves.mp4’. In ‘Pattern3.mp4’, surface ripples can be observed, which creates realistic visual effects. The reference videos of flag simulation are prefixed by “Reference-”. Compared with ‘Sand.mp4’, in ‘Sand\_All\_Gaussians.mp4’ the solid object is moving directly without surface deformation, which demonstrated the necessity of surface modeling.

## References

- [1] Chenfanfu Jiang, Craig Schroeder, Andrew Selle, Joseph Teran, and Alexey Stomakhin. The affine particle-in-cell method. *ACM Transactions on Graphics (TOG)*, 34(4):1–10, 2015. [1](#)
- [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*, New York, NY, USA, 2016. Association for Computing Machinery. [1](#)
- [3] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *TOG*, 42(4):1–14, 2023. [1](#)
- [4] Gergely Klár, Theodore Gast, Andre Pradhana, Chuyuan Fu, Craig Schroeder, Chenfanfu Jiang, and Joseph Teran. Drucker-prager elastoplasticity for sand animation. *ACM Transactions on Graphics (TOG)*, 35(4):1–12, 2016. [1](#)
- [5] Tianyi Xie, Zeshun Zong, Yuxing Qiu, Xuan Li, Yutao Feng, Yin Yang, and Chenfanfu Jiang. Physgaussian: Physics-integrated 3d gaussians for generative dynamics. In *CVPR*, pages 4389–4398, 2024. [1](#)
- [6] Zeshun Zong, Xuan Li, Minchen Li, Maurizio M Chiaramonte, Wojciech Matusik, Eitan Grinspun, Kevin Carlberg, Chenfanfu Jiang, and Peter Yichen Chen. Neural stress fields for reduced-order elastoplasticity and fracture. In *SIGGRAPH Asia 2023 Conference Papers*, pages 1–11, 2023. [1](#)