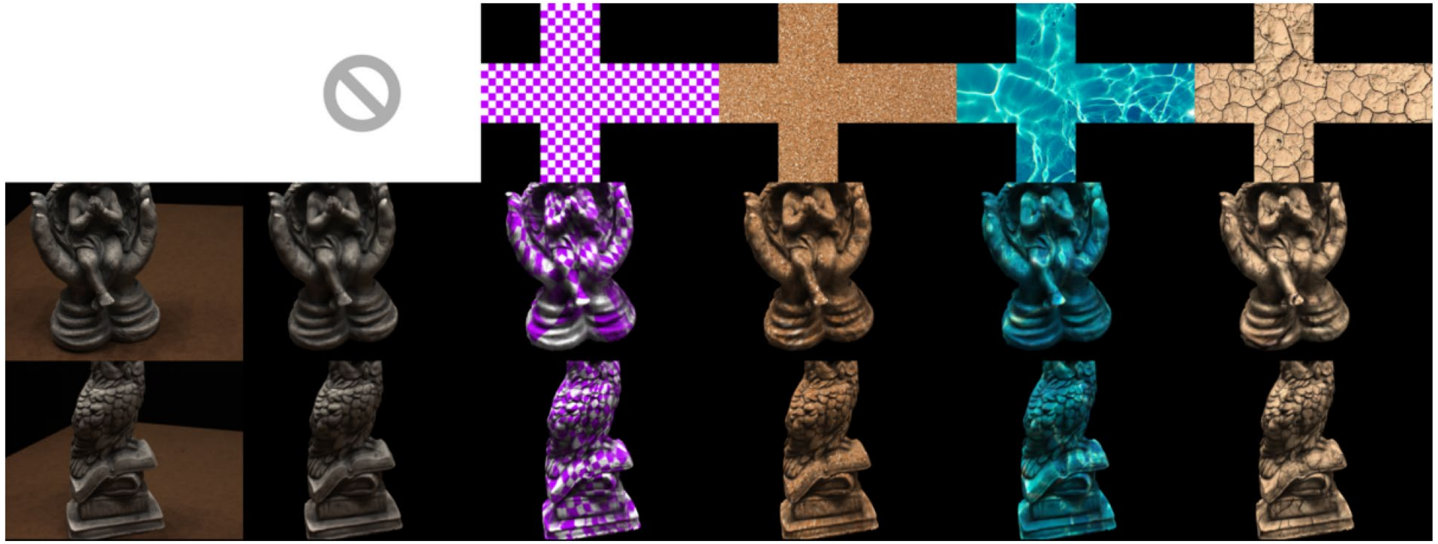
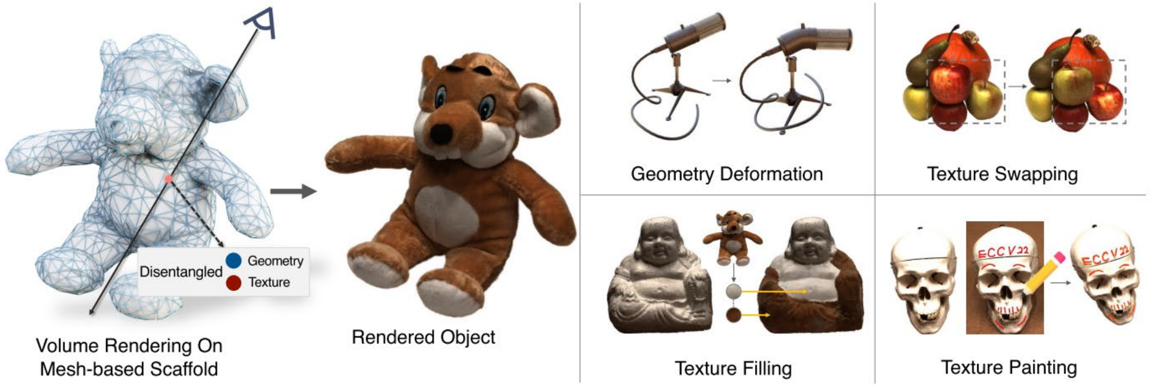
Texture-GS: Disentangle the Geometry and Texture for 3D Gaussian Splatting Editing

Tian-Xing Xu[1], Wenbo Hu[2], Yu-Kun Lai[3], Ying Shan[2] and Song-Hai Zhang[1]

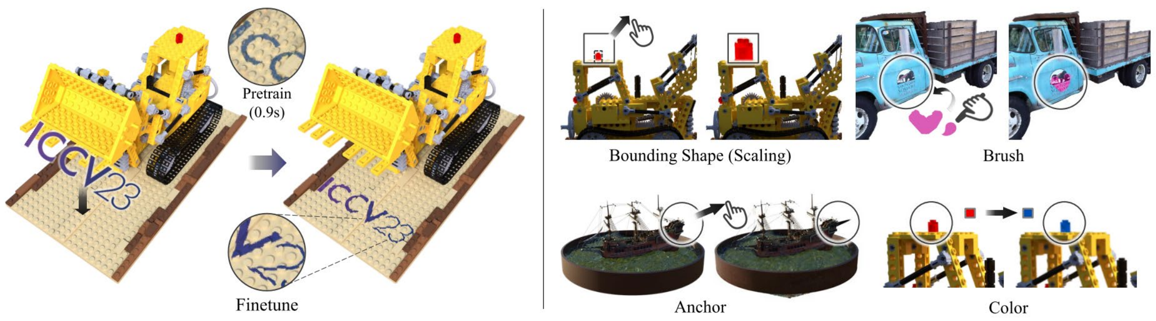
[1]Tsinghua University, [2]Tencent AI Lab, [3]Cardiff University



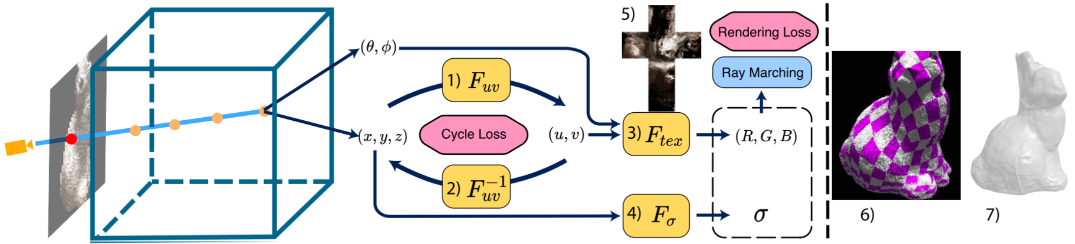
NeRF + Editing



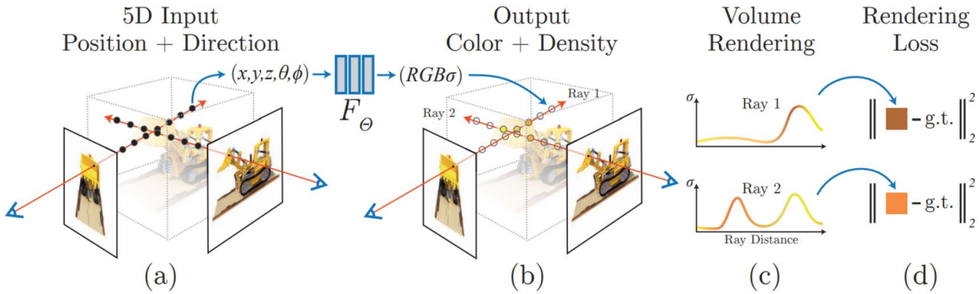
NeuMesh



Seal-3D



NeuTex



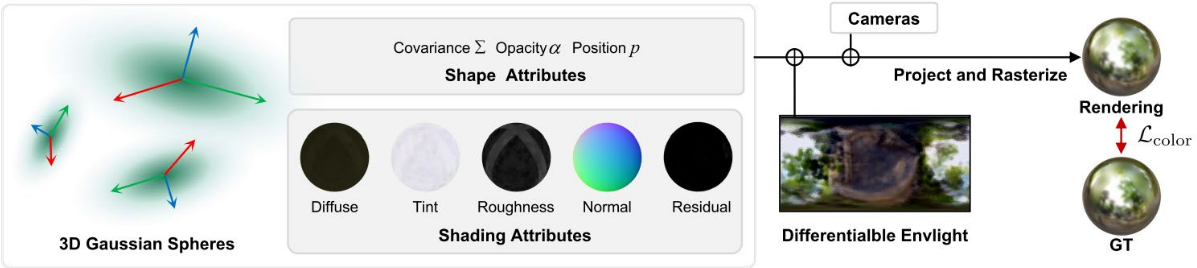
Volume Rendering

Slow! Cannot support real-time editing preview!

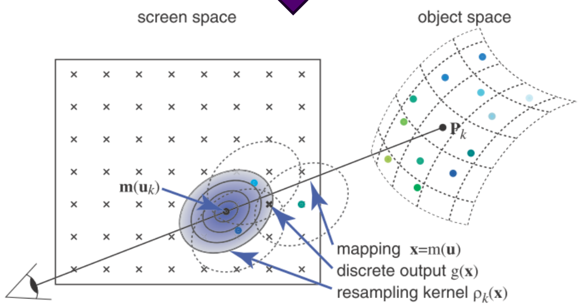
3D-GS + Editing

GaussianEditor

PhysGaussian



GaussianShader



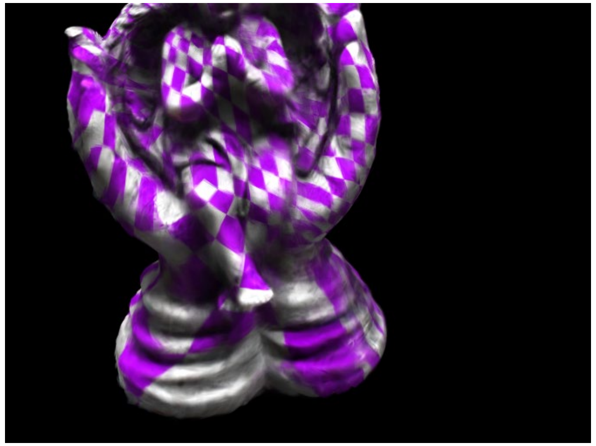
Surface Splatting

Entangle geometry (μ, σ, o) and texture (RGB/SH)

Can we disentangle the geometry and texture for 3D-GS like mesh?

Multi-view Images

Reconstruct



Geometry

ϕ: UV mapping

Texture

Application :

- Texture Painting
- Texture Swapping
- …

Geometry Reconstruction

- Geometry Reconstruction
  - Vanilla 3D-GS → Solid Surface
  - Opacity Regularization

$$\mathcal{L}_{01} = \frac{1}{N}\sum_{i=1}^{N}(\ln(o_i) + \ln(1 - o_i))$$

  - Normal Regularization

$$\mathcal{L}_{\text{normal}} = \frac{1}{HW}||\overline{N} - \overline{N}_{\text{gt}}||_2^2$$

$$\mathcal{L}_{\text{smooth}} = \frac{1}{HW}\sum_{p}\sum_{q\in\mathcal{N}(p)} \exp(-\gamma||C_{\text{gt}}(p) - C_{\text{gt}}(q)||_1)||\overline{N}(p) - \overline{N}(q)||_1$$

- UV Mapping Learning
- Texture Reconstruction

UV Mapping Learning

- Geometry Reconstruction
- UV Mapping Learning
  - Discard color attributes and freeze 3D Gaussian Parameters
  - Joint learn the mapping function ϕ: 3D → 2D and inverse mapping function ϕ^{-1}: 2D → 3D

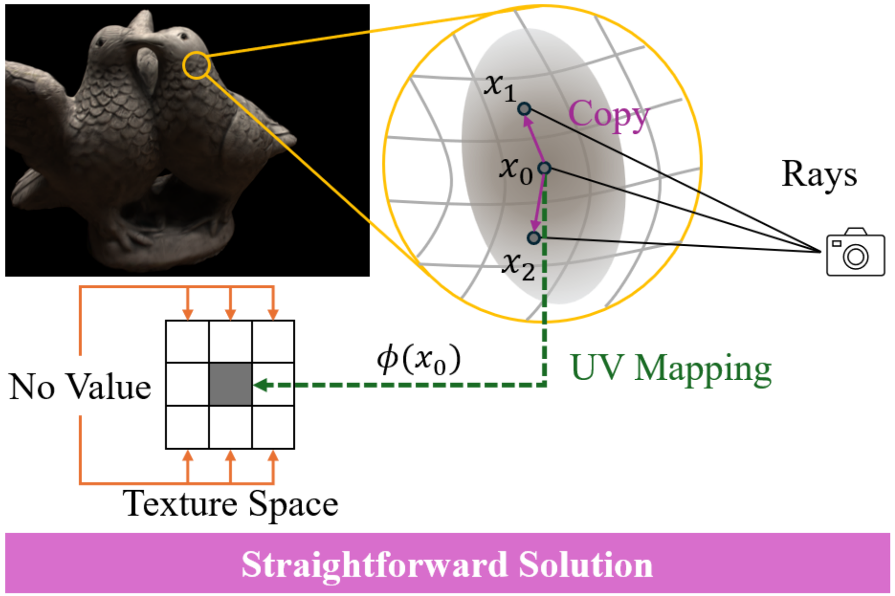
$$\mathcal{L}_{\text{cycle}} = \frac{1}{N_d}\sum_{i=1}^{N_d} ||x_i - \phi^{-1} \circ \phi(x_i)|| \quad \mathcal{L}_{\text{cycle2}} = \frac{1}{N_u}\sum_{i=1}^{N_u} ||u_i - \phi \circ \phi^{-1}(u_i)||$$

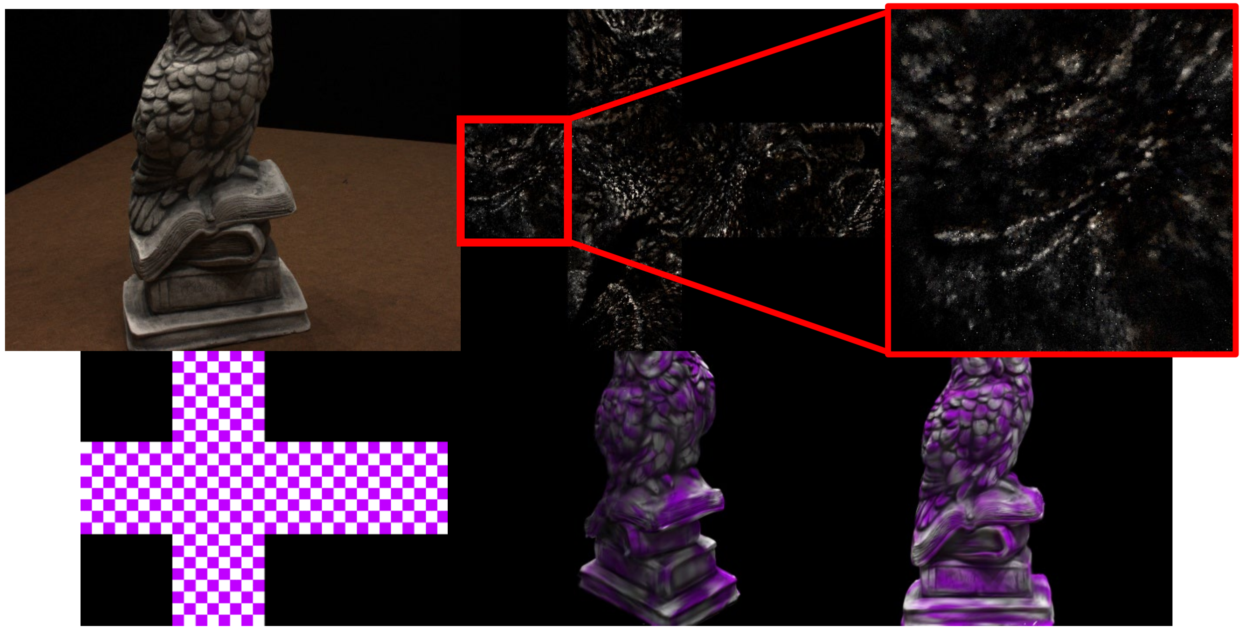
$$\mathcal{L}_{\text{CD}} = \frac{1}{N_u}\sum_{i=1}^{N_u} \min_{p_j \in \mathcal{P}} ||\phi^{-1}(u_i) - p_j|| + \frac{1}{N_p}\sum_{j=1}^{N_p} \min_{u_i \in \mathcal{U}} ||\phi^{-1}(u_i) - p_j||$$

  - Remove positional embedding to ensure the local continuity of the mapping function
  - Back-project rendered depth maps to obtain sampling points on the surface
- Texture Reconstruction

Texture Reconstruction

- Geometry Reconstruction
- UV Mapping Learning
- Texture Reconstruction
  - Naïve Solution : Pre-fetch color attributes from a learnable texture image before rendering





Texture Reconstruction

- Ours : Treat each 3D Gaussians as a surface (not a point) during rendering

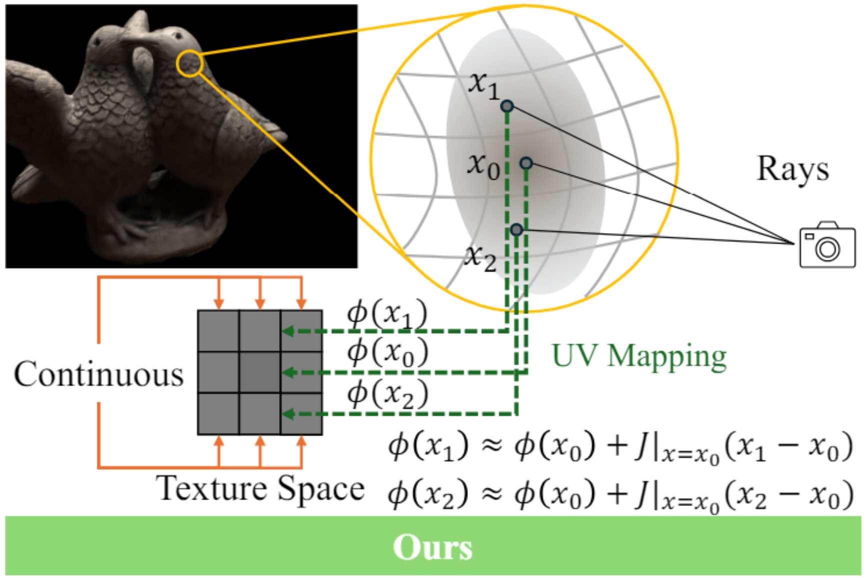
$$C_p = \sum_{j \in \mathcal{N}_p} c_j \alpha_j \prod_{k=1}^{j-1} (1 - \alpha_k), \quad \Rightarrow \quad C_p = \sum_{j \in \mathcal{N}_p} \mathcal{C}(G_j, r_p) \alpha_j \prod_{k=1}^{j-1} (1 - \alpha_k).$$

  - Ray-Gaussian Intersection

$$I(G_j, r_p) = o + \frac{(\mu_j - o) \cdot n_j}{d_p \cdot n_j} d_p.$$

  - Efficient UV Mapping

$$\tilde{\phi}(I(G_j, r_p)) = \phi(\mu_j) + J|_{x=\mu_j} (I(G_j, r_p) - \mu_j),$$



Texture-based 3D Gaussian Splatting

- Ours : Treat each 3D Gaussians as a surface (not a point) during rendering
  - Hybrid Color Representation for view-dependent appearance

$$C_p = \sum_{j \in \mathcal{N}_p} \mathcal{C}(G_j, r_p)\alpha_j \prod_{k=1}^{j-1}(1 - \alpha_k).$$

$$\mathcal{C}(G_j, r_p) = h(\tilde{\phi}(I(G_j, r_p)), \mathcal{T}) + c_j^{\mathrm{SH}},$$

Render Quality & Speed

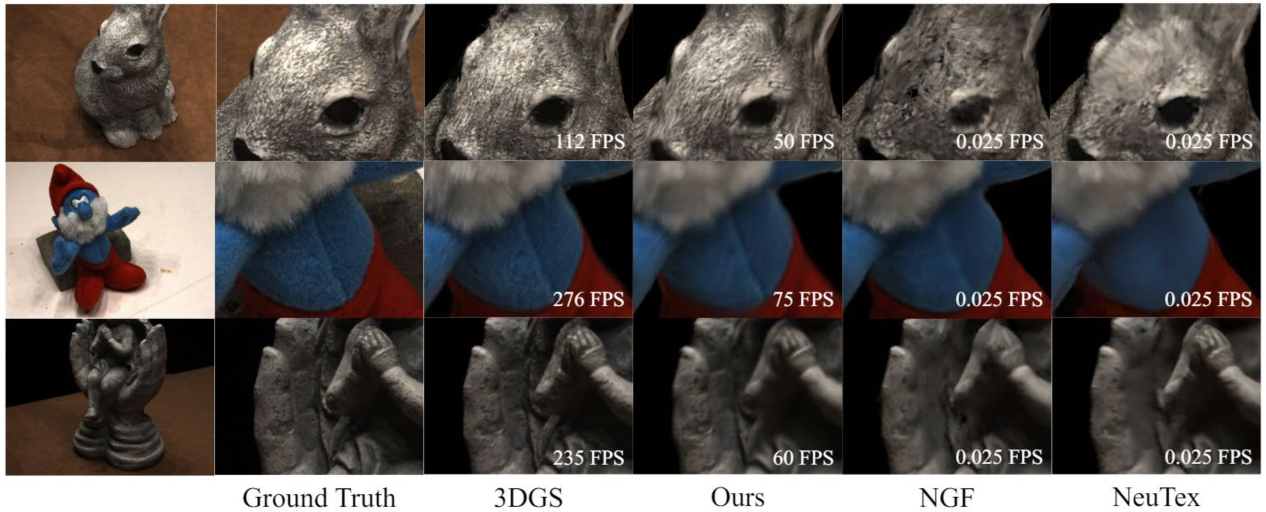


Fig. 3: Visual comparison with previous state-of-the-art editing methods

Table 1: Comparison of novel view synthesis results on the DTU dataset.

(a) Comparison with the SOTAs

Method	DTU			
	PSNR↑	L1↓	LPIPS↓	FPS
NeuTex	30.39	0.0158	0.1613	0.025
NGF	29.44	0.0166	0.1506	0.025
3DGS	30.99	0.0121	0.1079	198
Ours	30.03	0.0135	0.1440	58

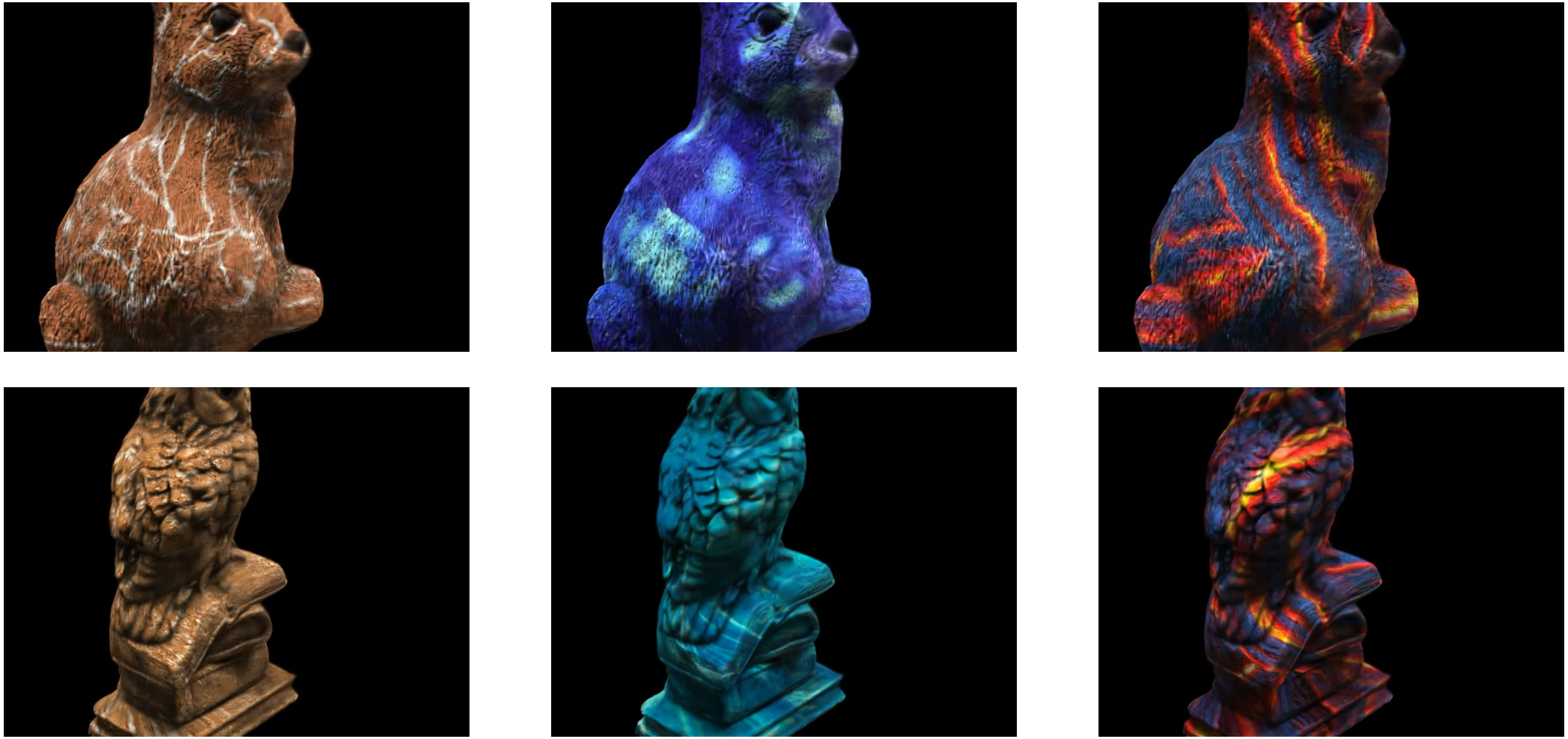
(b) Different number of 3D Gaussians

#Gauss	DTU			
	PSNR↑	L1↓	LPIPS↓	FPS
100%	30.03	0.0135	0.1440	58
50%	29.57	0.0142	0.1555	69
20%	28.75	0.0155	0.1705	82
5%	27.86	0.0172	0.1841	104

↑

Enhance the representation power of each 3D Gaussian

Texture Swapping



Texture Painting

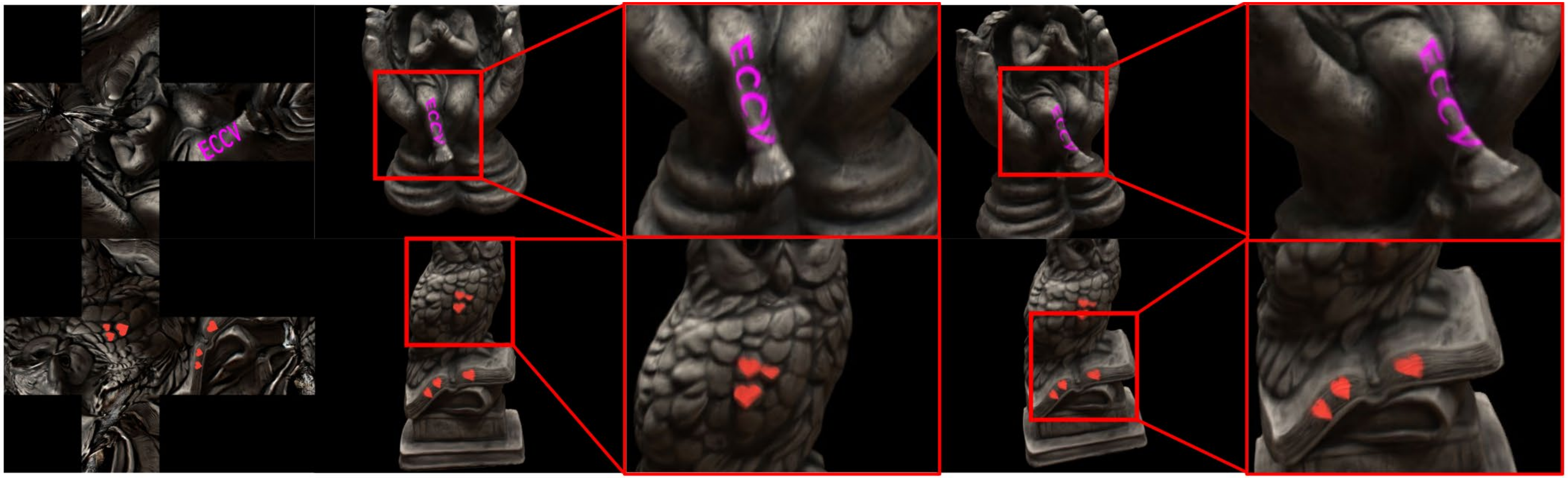
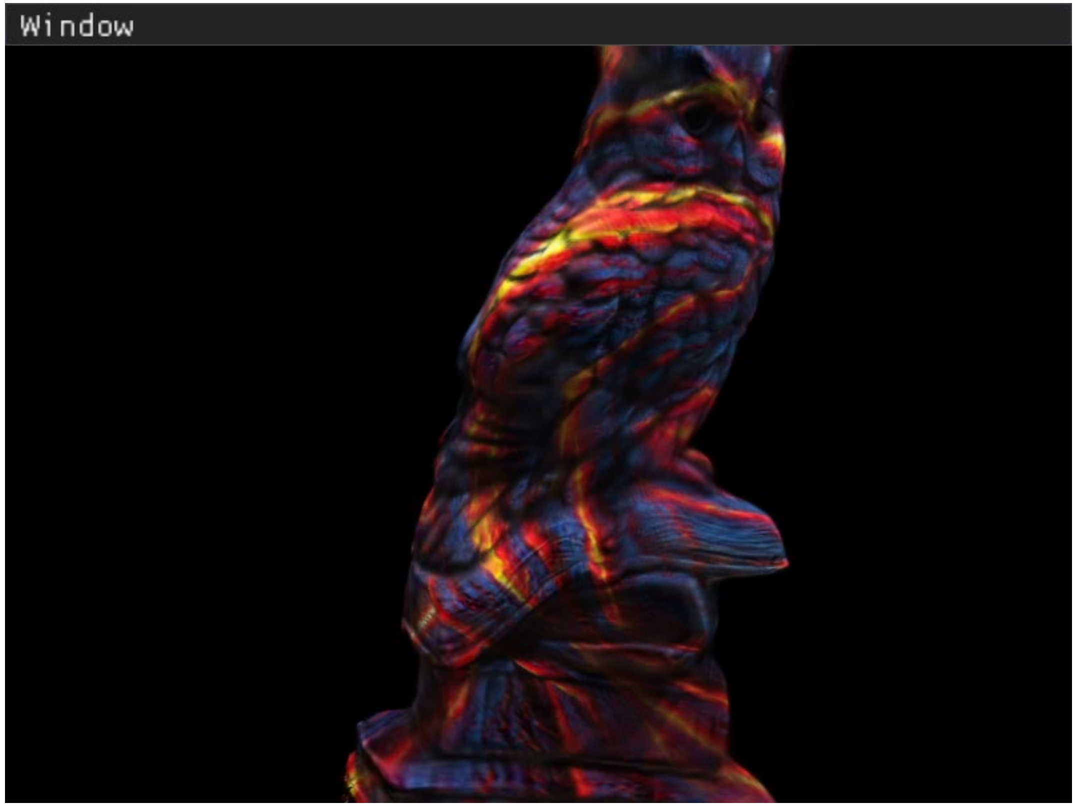
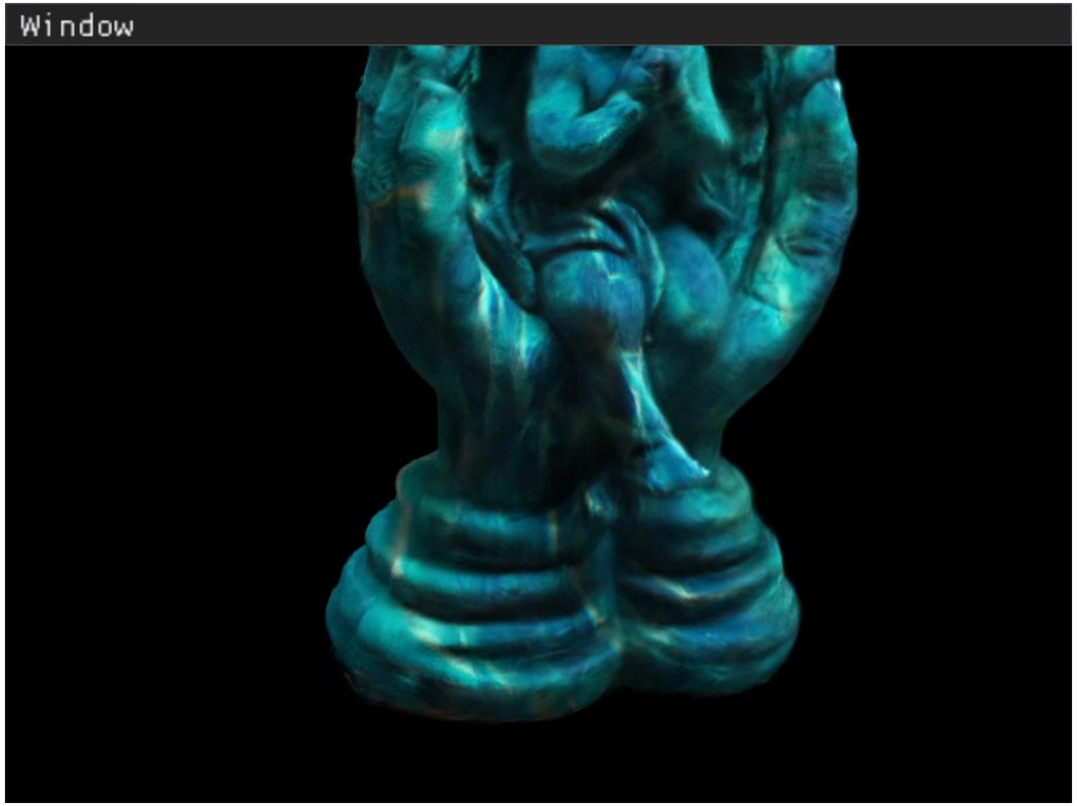


Fig. 7: Visualization of texture painting results of our method

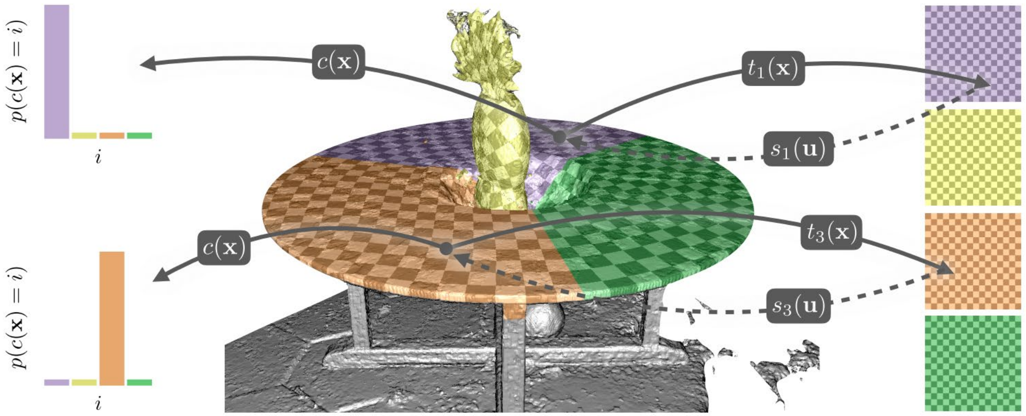
Real-time Rendering Speed for Instant Preview (RTX 2080Ti)





Conclusions

- We are the first to disentangle the geometry and texture for 3D-GS, thereby enabling various editing applications.
- Surface or Point : We are the first to treat each 3D Gaussian as a shading surface (not a shading point in NeRF) during rendering.
- Limitations :
  - Rely on the accuracy of the learned UV mapping function
  - Unconstrained UV mapping learning on objects of the same category (human faces, …)



Future Works

- UV Mapping Learning
  - Objects of the same category (human faces, human bodies, …)
  - More accurate UV mapping
- 3D AIGC
  - Fewer 3D Gaussians for objects with simple geometry and rich texture

Welcome to contact me for further discussion!

😊 Tian-Xing Xu 😊
xutx21@mails.tsinghua.edu.cn

Code : https://github.com/slothfulxtx/Texture-GS
Demo page : https://slothfulxtx.github.io/TexGS/