更优3D表征方法

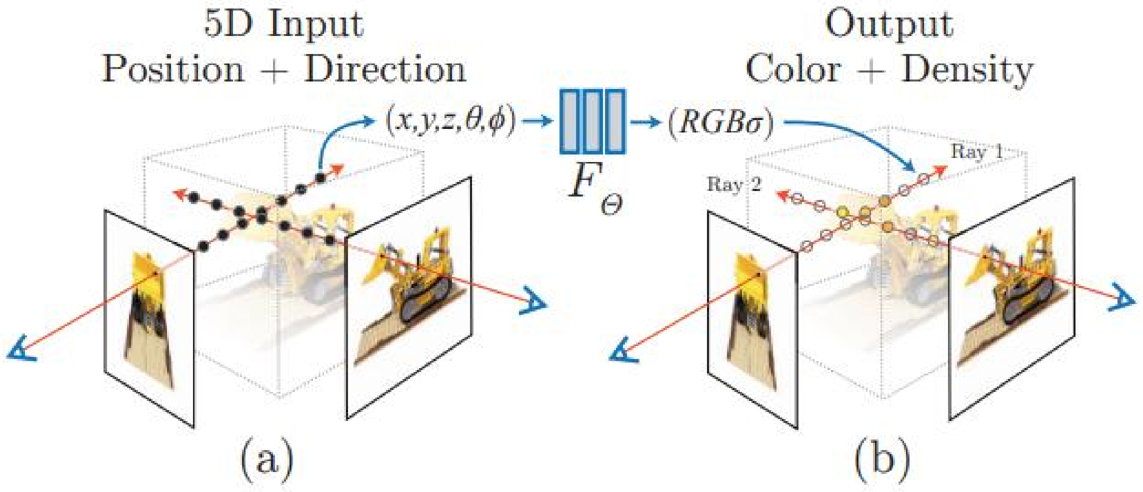
- Tetsphere Splatting
- 3D Shape Tokenization

[1] Guo M, Wang B, He K, et al. TetSphere Splatting: Representing High-Quality Geometry with Lagrangian Volumetric Meshes. ICLR, 2025.
[2] Chang J H R, Wang Y, Martin M A B, et al. 3D Shape Tokenization. arXiv preprint arXiv:2412.15618, 2024.

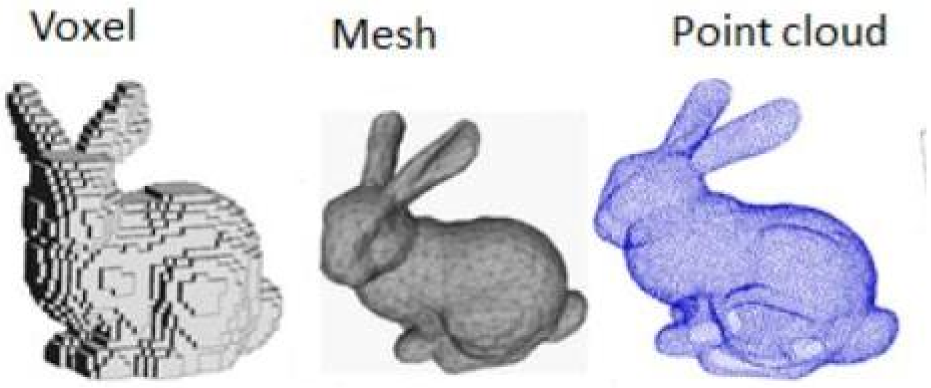
大背景

现有的几何表征方法可以分为两类：隐式表征和显式表征。

- 隐式表征方法使用神经网络来拟合场景中的光照分布，代表方法是NeRF，SDF。
- 显式表征方法使用离散的基元来表征场景，每个基元存储对应的光照信息。代表方法有：体素，网格，点云，3DGS等。



NeRF



显式表征

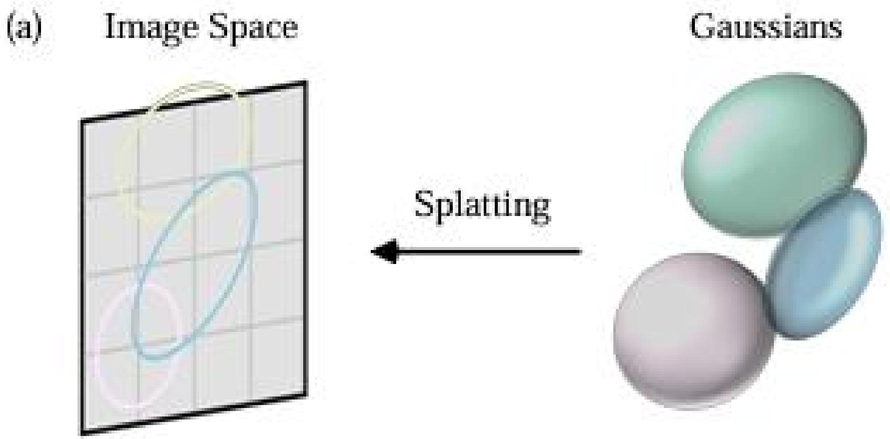
前情提要：3DGS

区别于点云，3DGS用三维椭球作为基元来显式表征场景。

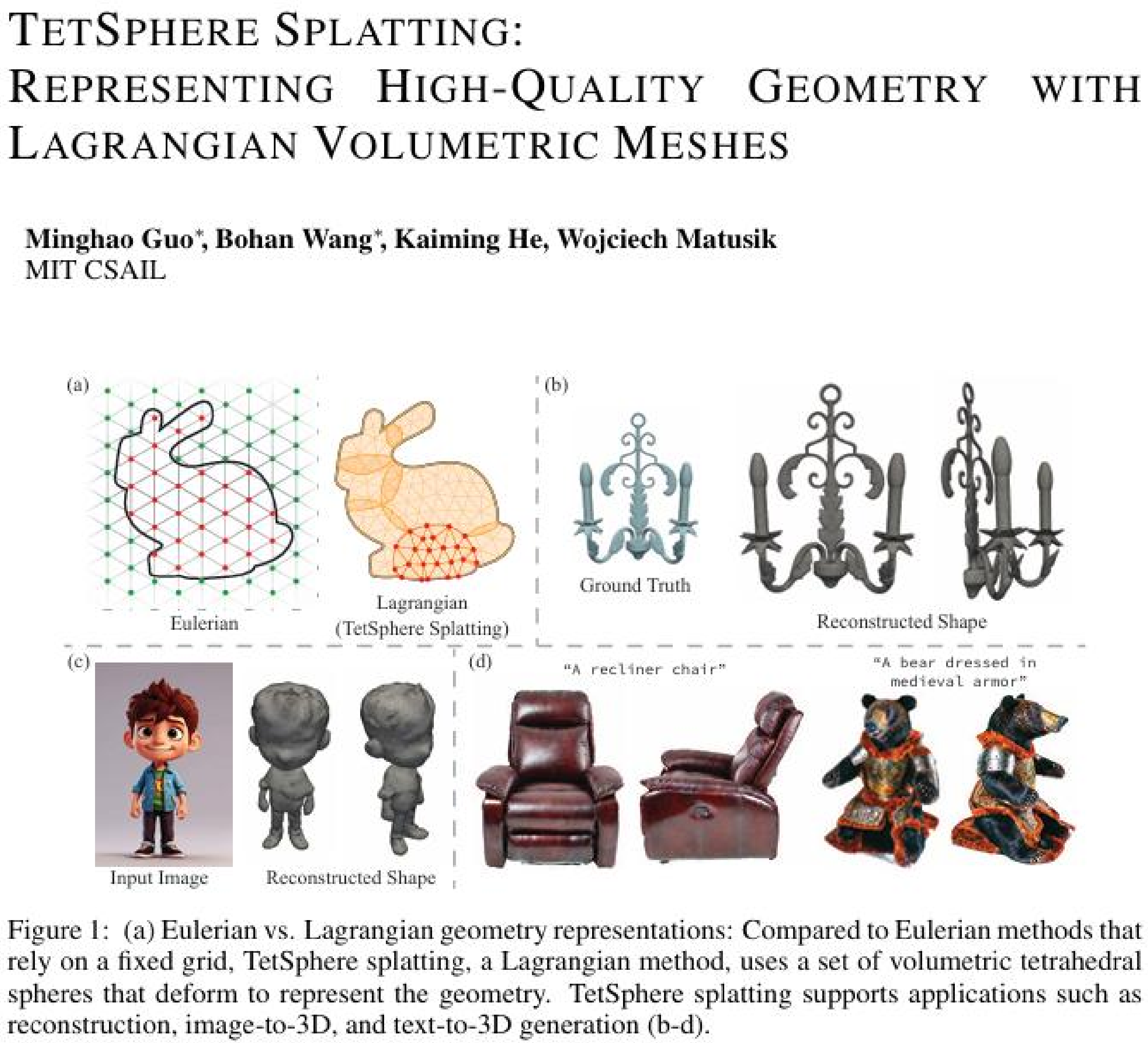
一个大型场景通常需要100K以上椭球来建模。

- 每个椭球参数：
  - ➢ 三维坐标x,y,z
  - ➢ 协方差矩阵∑（决定椭球形状和方向）
  - ➢ 不透明度α
  - ➢ 球谐函数参数sh（拟合颜色）

- 渲染过程可以理解将三维椭球抛到成像平面上，根据椭球的前后顺序，累积得到像素颜色。



1、ICLR’25 TETSPHERE SPLATTING



TetSphere Splatting: Representing High-Quality Geometry with Lagrangian Volumetric Meshes

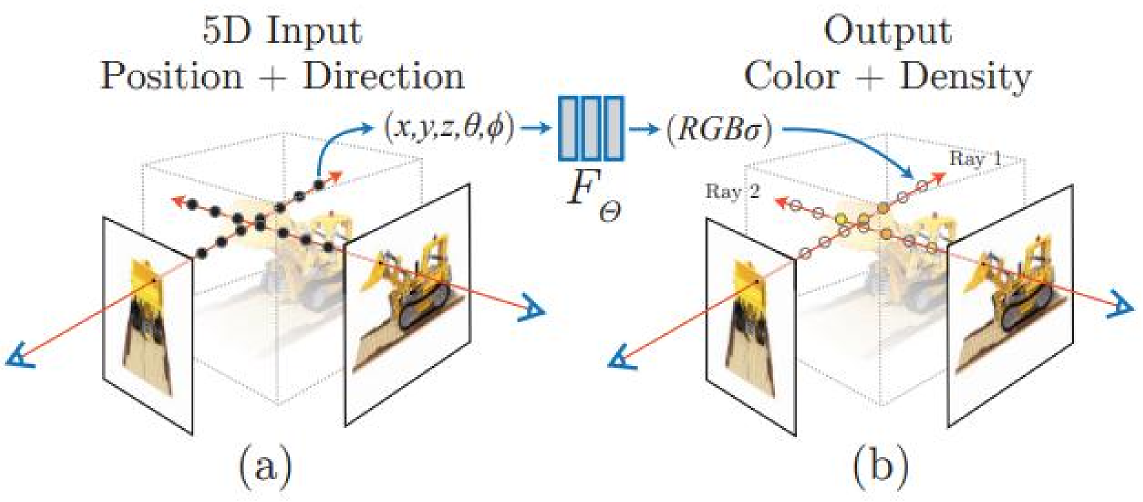
Minghao Guo*, Bohan Wang*, Kaiming He, Wojciech Matusik
MIT CSAIL

Figure 1: (a) Eulerian vs. Lagrangian geometry representations: Compared to Eulerian methods that rely on a fixed grid, TetSphere splatting, a Lagrangian method, uses a set of volumetric tetrahedral spheres that deform to represent the geometry. TetSphere splatting supports applications such as reconstruction, image-to-3D, and text-to-3D generation (b-d).

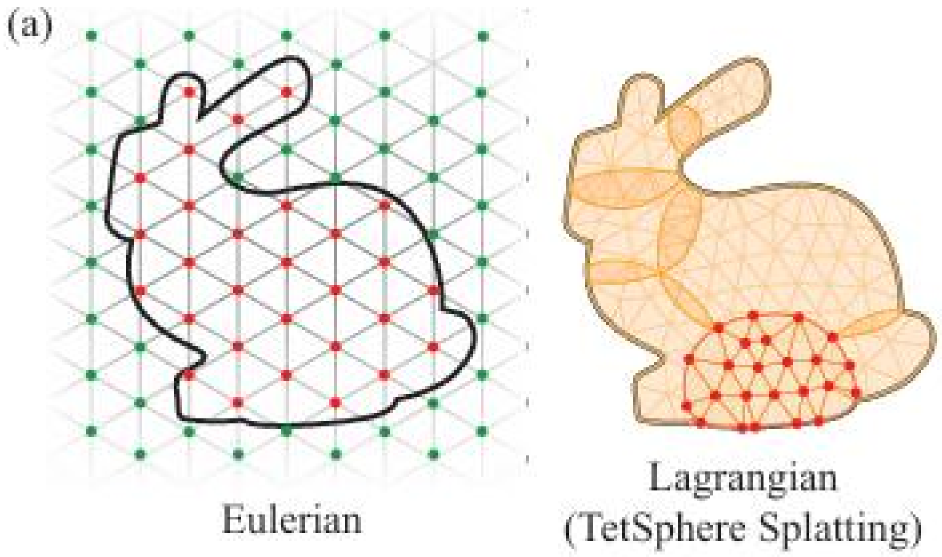
背景

现有的几何表征方法可以分为两类：欧拉法和拉格朗日法。

- 欧拉法依赖固定的坐标，随着分辨率增加，计算消耗也会大幅增加。代表有NeRF、FlexiCubes等。
- 拉格朗日法有更好的计算效率，但是在几何质量上通常更差。代表方法有3DGS，DMesh等。



NeRF



挑战

建模精度有待提升

- 欧拉法依赖固定的坐标，难以捕捉一些细长结构。
- 拉格朗日优化每个点或者网格，缺少整体结构的一致性。

→ 不利于下游任务 (渲染、仿真) 的发展

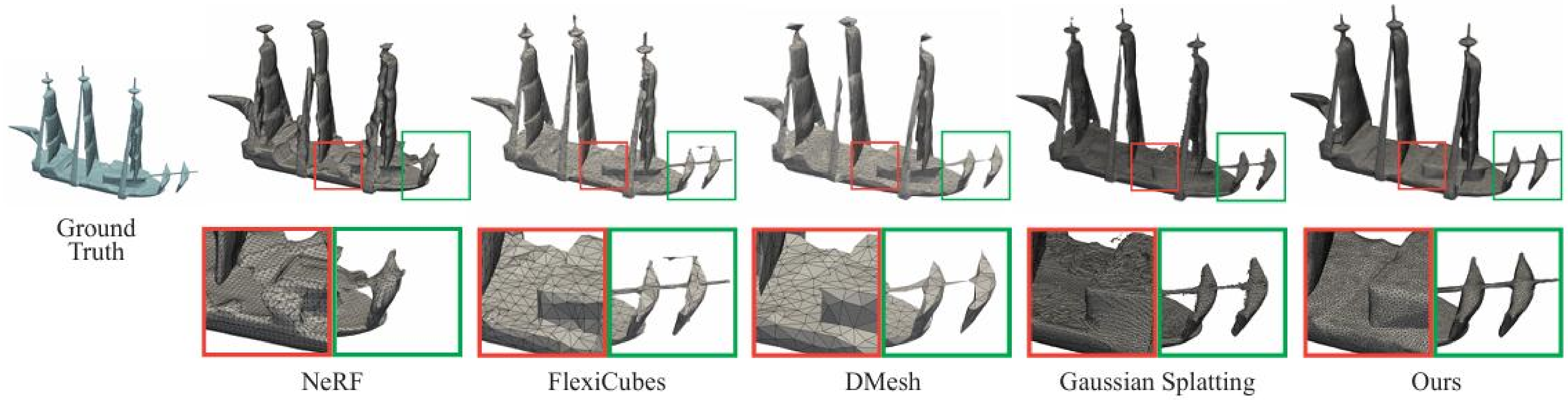
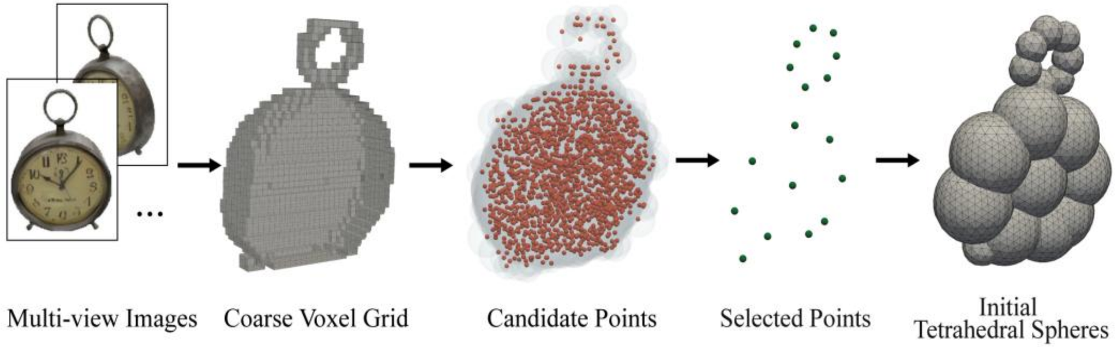


Figure 2: Visual comparison of mesh quality across widely used shape representations, including NeRF (Mildenhall et al., 2020), FlexiCubes (Shen et al., 2023b) (Eulerian), DMesh (Son et al., 2024), and Gaussian Splatting (Huang et al., 2024) (Lagrangian). These methods exhibit mesh quality issues, such as irregular or degenerated triangles, non-manifoldness, and floating artifacts. Our method demonstrates uniform surface triangles, improved mesh quality, and structure integrity.

方法

- **使用四面体球作为基元。**相比于点云，由于存在四面体的结构化约束，保持了物体的几何完整性。
- 从一组四面体球开始，优化顶点的位置，使得渲染图像与相应的目标图像对齐（渲染损失）。顶点的运动受到四面体表面的正则化约束，惩罚非光滑变形。



初始化

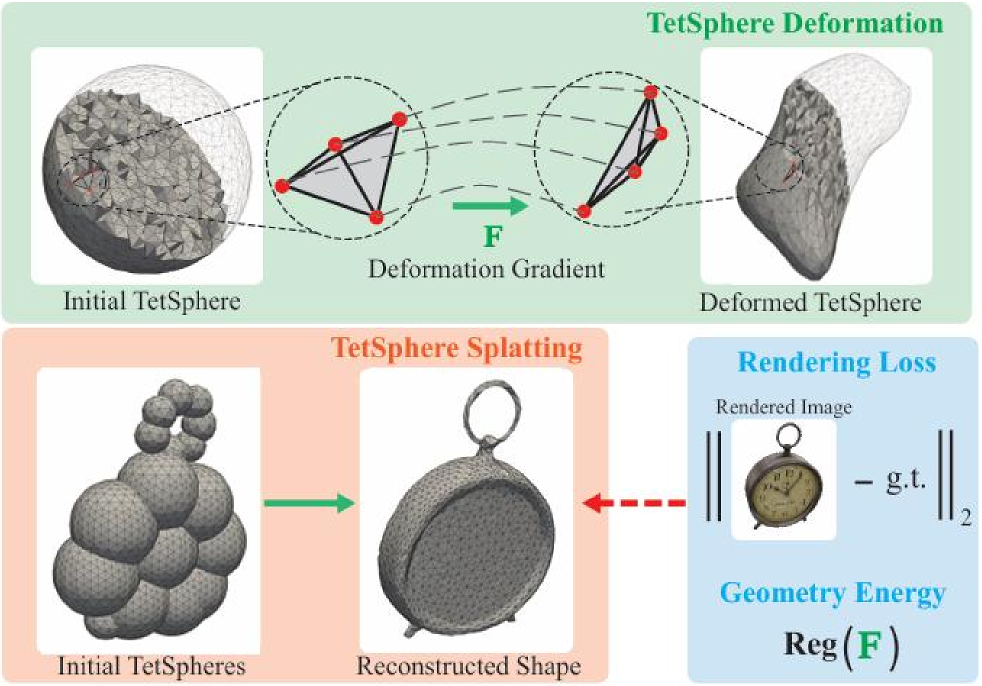


Figure 3: Overall pipeline: TetSphere splatting represents a 3D shape using a collection of TetSpheres. Each TetSphere is a tetrahedral sphere that can be deformed from its initial uniform state through deformation gradient. The deformation process is optimized by minimizing rendering loss and geometric energy terms.

方法

正则化

- 引入双谐波能量[1]，惩罚顶点相对位置剧烈变化，保证平滑，减少物体表面的不合理波动。
- 引入几何约束[2]，确保四面体保持正体积，避免顶点位移时发生翻转和坍塌现象。

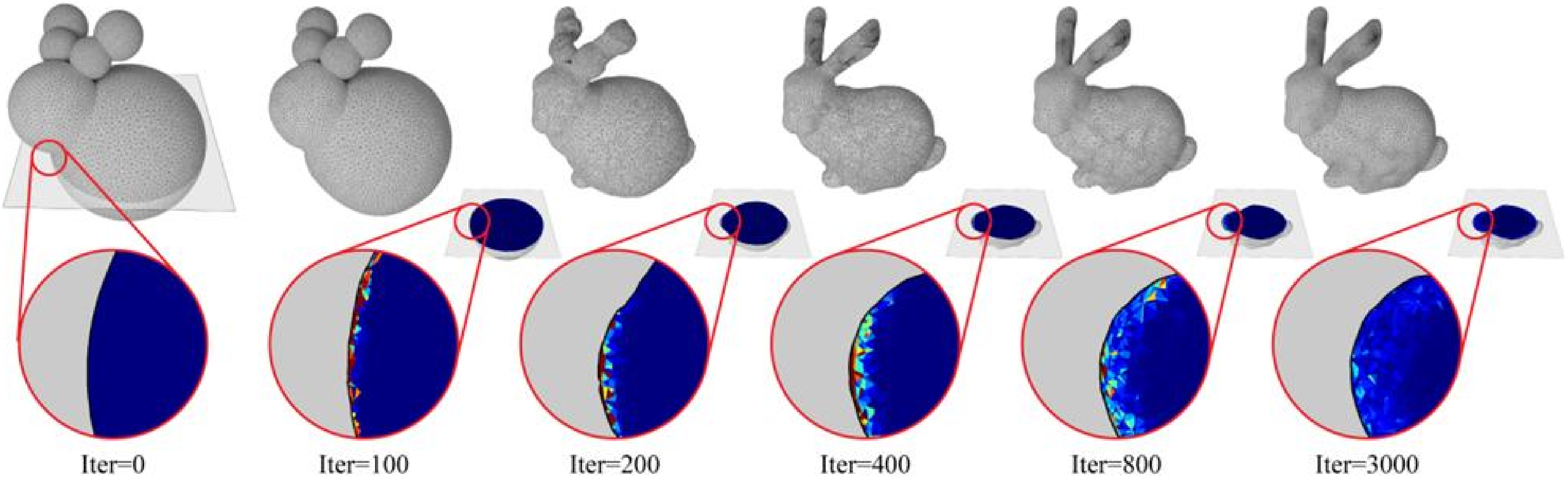


Figure 4: TetSphere splatting with deforming tetrahedral spheres. Color-coded regions represent the bi-harmonic energy values (red: high, blue: low) across tetrahedra, one of the geometric regularizations employed in our deformation optimization process.

[1] Mario Botsch and Olga Sorkine. On linear variational surface deformation methods. IEEE transactions on visualization and CG, 2007.
[2] Christian Schuller, Ladislav Kavan, Daniele Panozzo, and Olga Sorkine-Hornung. Locally injective mappings. In Computer Graphics Forum, 2013.

实验结果

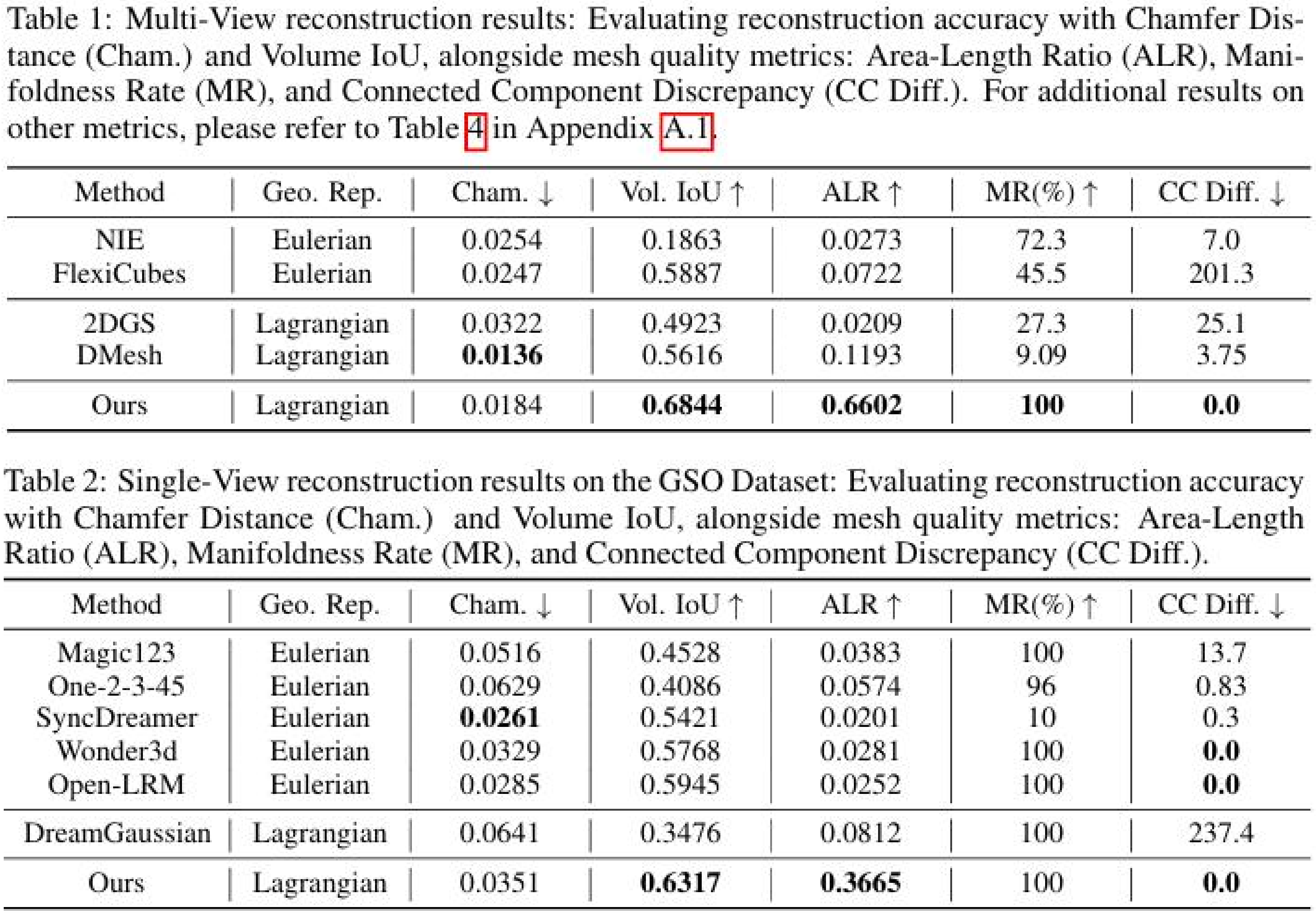


Table 1: Multi-View reconstruction results: Evaluating reconstruction accuracy with Chamfer Distance (Cham.) and Volume IoU, alongside mesh quality metrics: Area-Length Ratio (ALR), Manifoldness Rate (MR), and Connected Component Discrepancy (CC Diff.). For additional results on other metrics, please refer to Table 4 in Appendix A.1.

Method	Geo. Rep.	Cham. ↓	Vol. IoU ↑	ALR ↑	MR(%) ↑	CC Diff. ↓
NIE	Eulerian	0.0254	0.1863	0.0273	72.3	7.0
FlexiCubes	Eulerian	0.0247	0.5887	0.0722	45.5	201.3
2DGS	Lagrangian	0.0322	0.4923	0.0209	27.3	25.1
DMesh	Lagrangian	**0.0136**	0.5616	0.1193	9.09	3.75
Ours	Lagrangian	0.0184	**0.6844**	**0.6602**	**100**	**0.0**

Table 2: Single-View reconstruction results on the GSO Dataset: Evaluating reconstruction accuracy with Chamfer Distance (Cham.) and Volume IoU, alongside mesh quality metrics: Area-Length Ratio (ALR), Manifoldness Rate (MR), and Connected Component Discrepancy (CC Diff.).

Method	Geo. Rep.	Cham. ↓	Vol. IoU ↑	ALR ↑	MR(%) ↑	CC Diff. ↓
Magic123	Eulerian	0.0516	0.4528	0.0383	100	13.7
One-2-3-45	Eulerian	0.0629	0.4086	0.0574	96	0.83
SyncDreamer	Eulerian	**0.0261**	0.5421	0.0201	10	0.3
Wonder3d	Eulerian	0.0329	0.5768	0.0281	100	**0.0**
Open-LRM	Eulerian	0.0285	0.5945	0.0252	100	**0.0**
DreamGaussian	Lagrangian	0.0641	0.3476	0.0812	100	237.4
Ours	Lagrangian	0.0351	**0.6317**	**0.3665**	100	**0.0**

实验结果

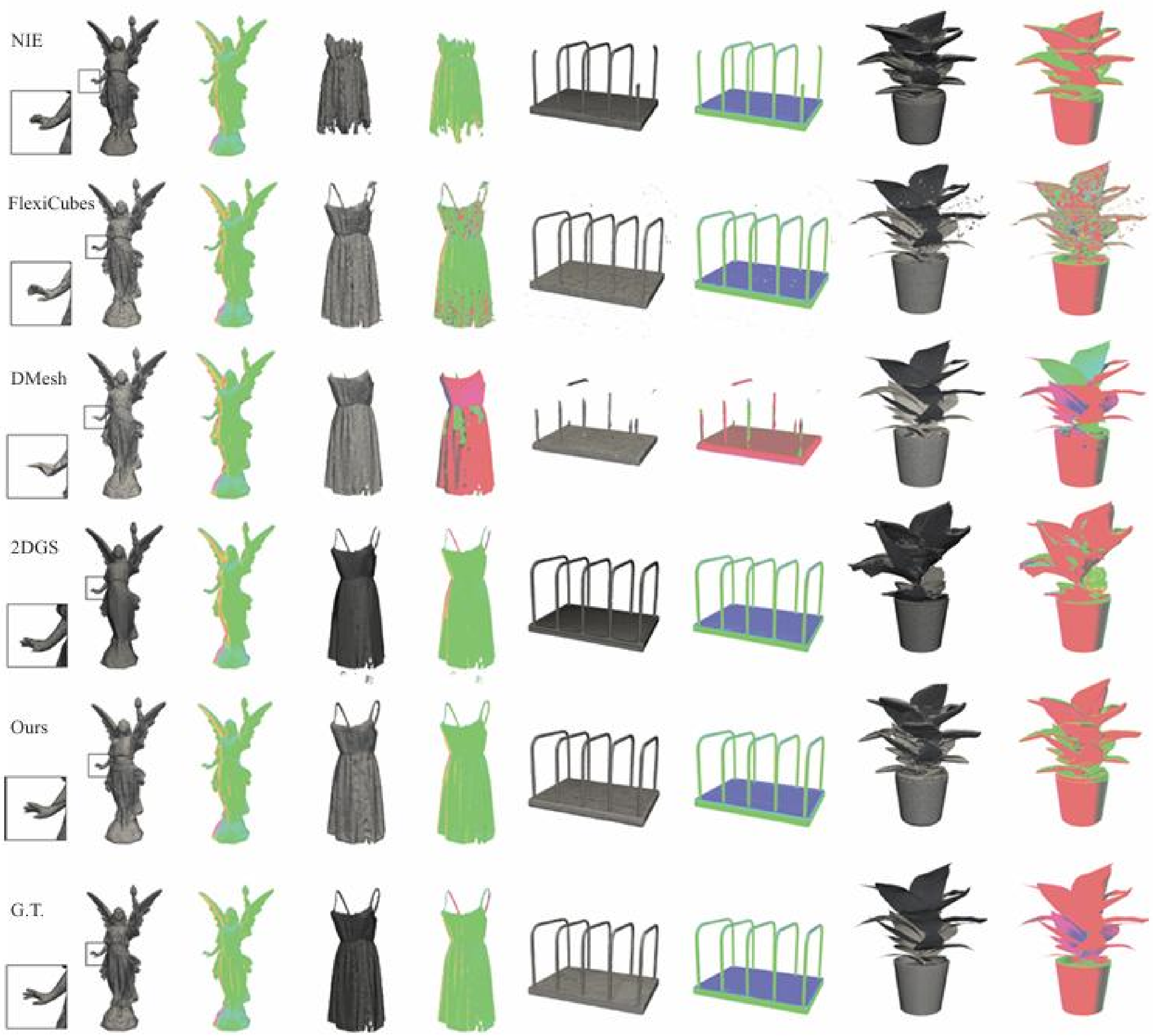


Figure 5: Qualitative results on multi-view reconstruction, with surface mesh visualizations and rendered normal maps. Our method excels over baseline methods regarding mesh quality, less bumpy surface, correct surface orientation, and accurately capturing slender and thin structures.

实验结果

- 相对不足：四面体参数化相对于2DGS和FlexiCubes更复杂，需要更长的优化时间。

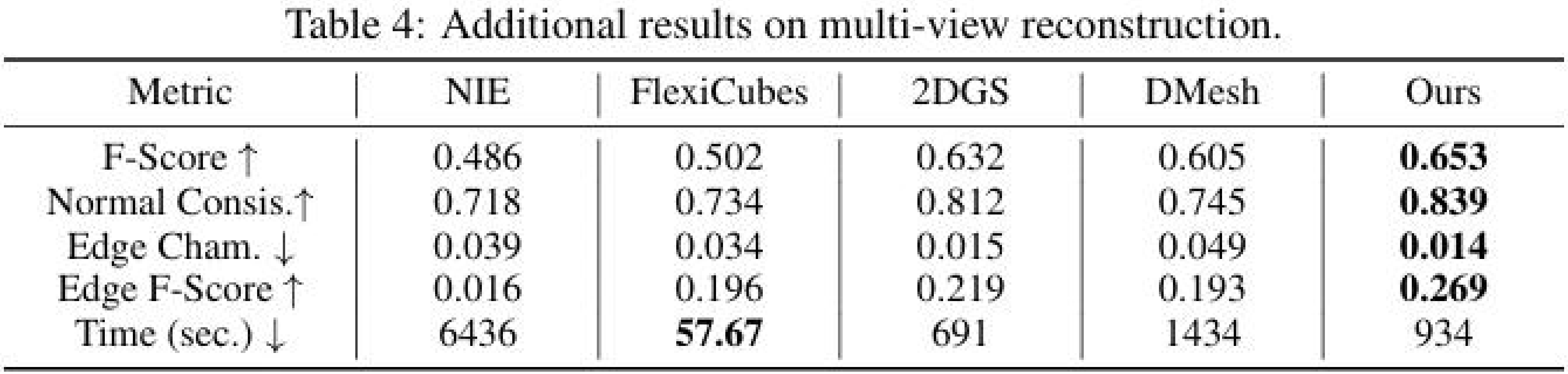
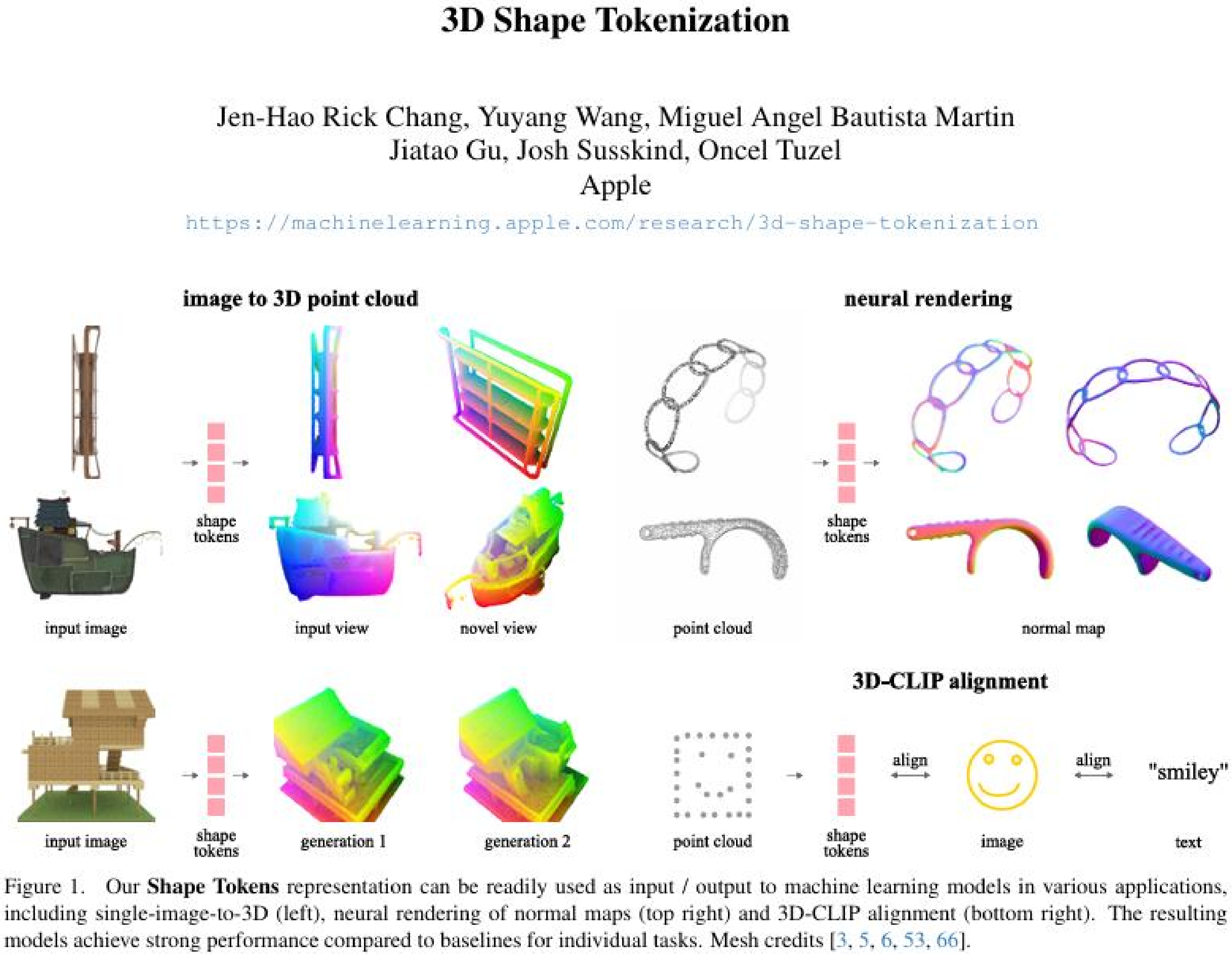


Table 4: Additional results on multi-view reconstruction.

Metric	NIE	FlexiCubes	2DGS	DMesh	Ours
F-Score ↑	0.486	0.502	0.632	0.605	**0.653**
Normal Consis.↑	0.718	0.734	0.812	0.745	**0.839**
Edge Cham. ↓	0.039	0.034	0.015	0.049	**0.014**
Edge F-Score ↑	0.016	0.196	0.219	0.193	**0.269**
Time (sec.) ↓	6436	**57.67**	691	1434	934

2、3D Shape Tokenization



3D Shape Tokenization

Jen-Hao Rick Chang, Yuyang Wang, Miguel Angel Bautista Martin
Jiatao Gu, Josh Susskind, Oncel Tuzel
Apple
https://machinelearning.apple.com/research/3d-shape-tokenization

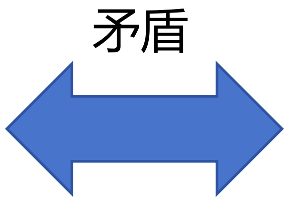
Figure 1. Our **Shape Tokens** representation can be readily used as input / output to machine learning models in various applications, including single-image-to-3D (left), neural rendering of normal maps (top right) and 3D-CLIP alignment (bottom right). The resulting models achieve strong performance compared to baselines for individual tasks. Mesh credits [3, 5, 6, 53, 66].

通用表征方法

在机器学习中，如何表示三维形状?

- 目前有很多表示方法，比如体素、网格、点云、符号距离函数等，但没有一种方法能适用于所有任务。

大多数机器学习模型需要
连续且紧凑的表征



现有的方法要么不够连续紧凑
（点云、网格等显式表征）
要么假设太多（如SDF要求形状是封闭的）

创新点

- 提出一种通用的三维形状表征方法（Shape Tokens，缩写ST），能够适用于多种机器学习任务。

优势

- 连续且紧凑。用少量向量（如 1024 个 16 维向量）表示复杂形状。
- 对三维形状最少的假设。SDF假设封闭形状，3DGS假设了体渲染。
- 只需要点云进行训练。现有方法需要网格或者符号距离函数来训练。
- 多功能：支持形状分析（如法线估计、去噪、变形等）。

方法

- 核心思想：将三维形状看作三维空间中的概率密度函数，通过流匹配生成模型（flow matching generative model）学习这些密度函数的表示。

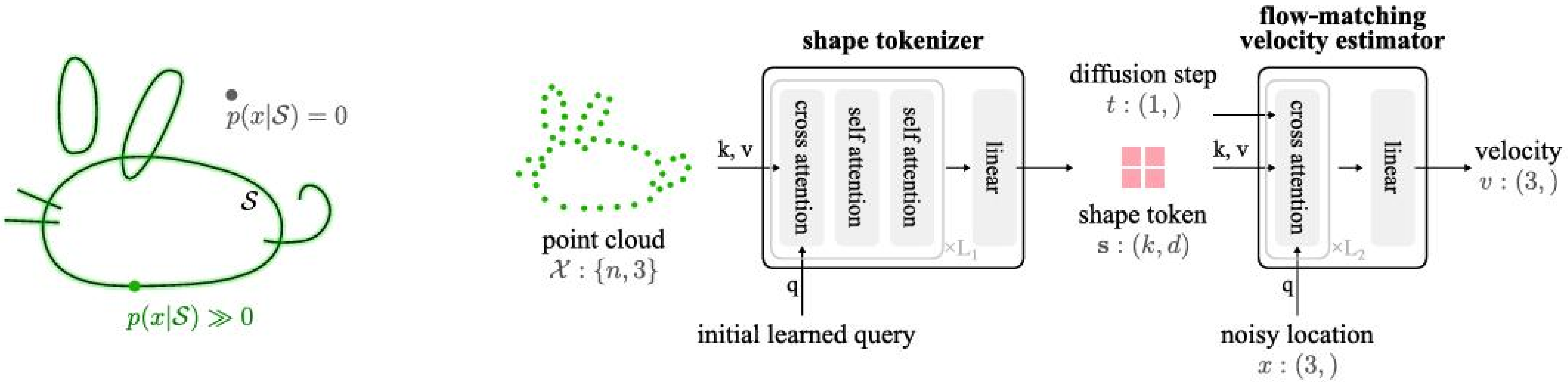
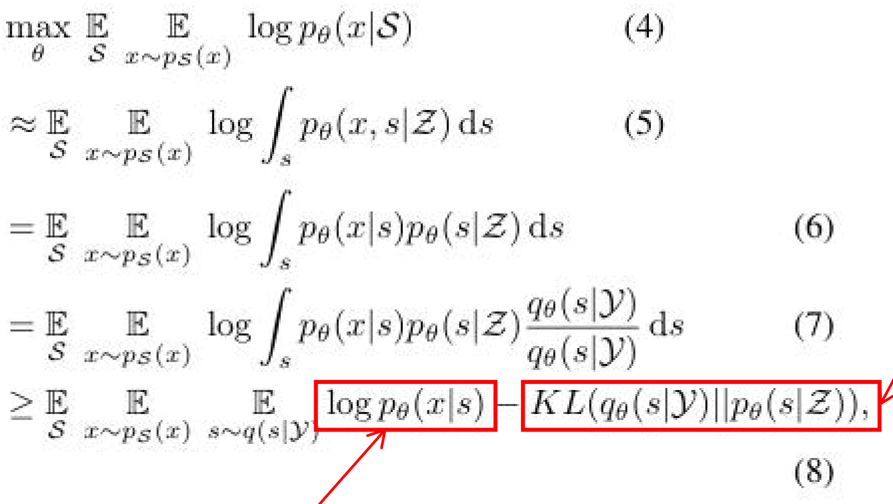


Figure 2. Overview of our architecture. (Left) We model a 3D shape as a probability density function that is concentrated on the surface, forming a delta function in 3D. (Right) Our tokenizer uses cross attention to aggregate information about the point cloud sampled on the shape into ST. The velocity estimator only use cross attention and MLP to maintain independence between points.

目标函数

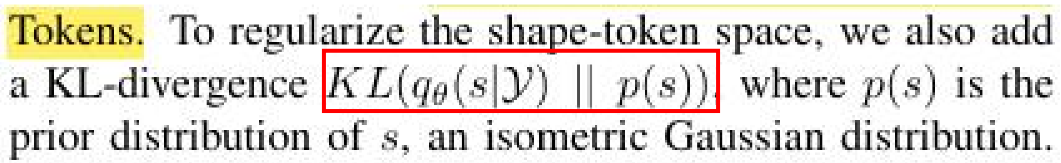
- 目的：训练Shape Tokenizer学会从点云中提取Shape Tokens，训练流匹配模型用ST调整点的位置(形状）。
- 目标函数：最大化对数似然的变分下限，让预测点的分布和真实点的分布接近。



$$
\begin{aligned}
&\max_{\theta} \mathop{\mathbb{E}}_{\mathcal{S}} \mathop{\mathbb{E}}_{x\sim p_{\mathcal{S}}(x)} \log p_\theta(x|\mathcal{S}) && (4)\\
&\approx \mathop{\mathbb{E}}_{\mathcal{S}} \mathop{\mathbb{E}}_{x\sim p_{\mathcal{S}}(x)} \log \int_s p_\theta(x,s|\mathcal{Z})\,\mathrm{d}s && (5)\\
&= \mathop{\mathbb{E}}_{\mathcal{S}} \mathop{\mathbb{E}}_{x\sim p_{\mathcal{S}}(x)} \log \int_s p_\theta(x|s)p_\theta(s|\mathcal{Z})\,\mathrm{d}s && (6)\\
&= \mathop{\mathbb{E}}_{\mathcal{S}} \mathop{\mathbb{E}}_{x\sim p_{\mathcal{S}}(x)} \log \int_s p_\theta(x|s)p_\theta(s|\mathcal{Z})\frac{q_\theta(s|\mathcal{Y})}{q_\theta(s|\mathcal{Y})}\,\mathrm{d}s && (7)\\
&\geq \mathop{\mathbb{E}}_{\mathcal{S}} \mathop{\mathbb{E}}_{x\sim p_{\mathcal{S}}(x)} \mathop{\mathbb{E}}_{s\sim q(s|\mathcal{Y})} \boxed{\log p_\theta(x|s)} - \boxed{KL(q_\theta(s|\mathcal{Y})||p_\theta(s|\mathcal{Z}))}, && (8)
\end{aligned}
$$

ST为s时，x在某位置的概率分布。由流匹配模型预测。

- 希望从同一个三维形状 S 中采样的两个点云Y和 Z生成的 Shape Tokens 是相似的。用 KL 散度来度量两个ST的相似性。
- 防止过拟合 (只记住训练数据，不会推广到新数据)



Tokens. To regularize the shape-token space, we also add a KL-divergence $\boxed{KL(q_\theta(s|\mathcal{Y})\ ||\ p(s))}$ where $p(s)$ is the prior distribution of s, an isometric Gaussian distribution.

实验结果

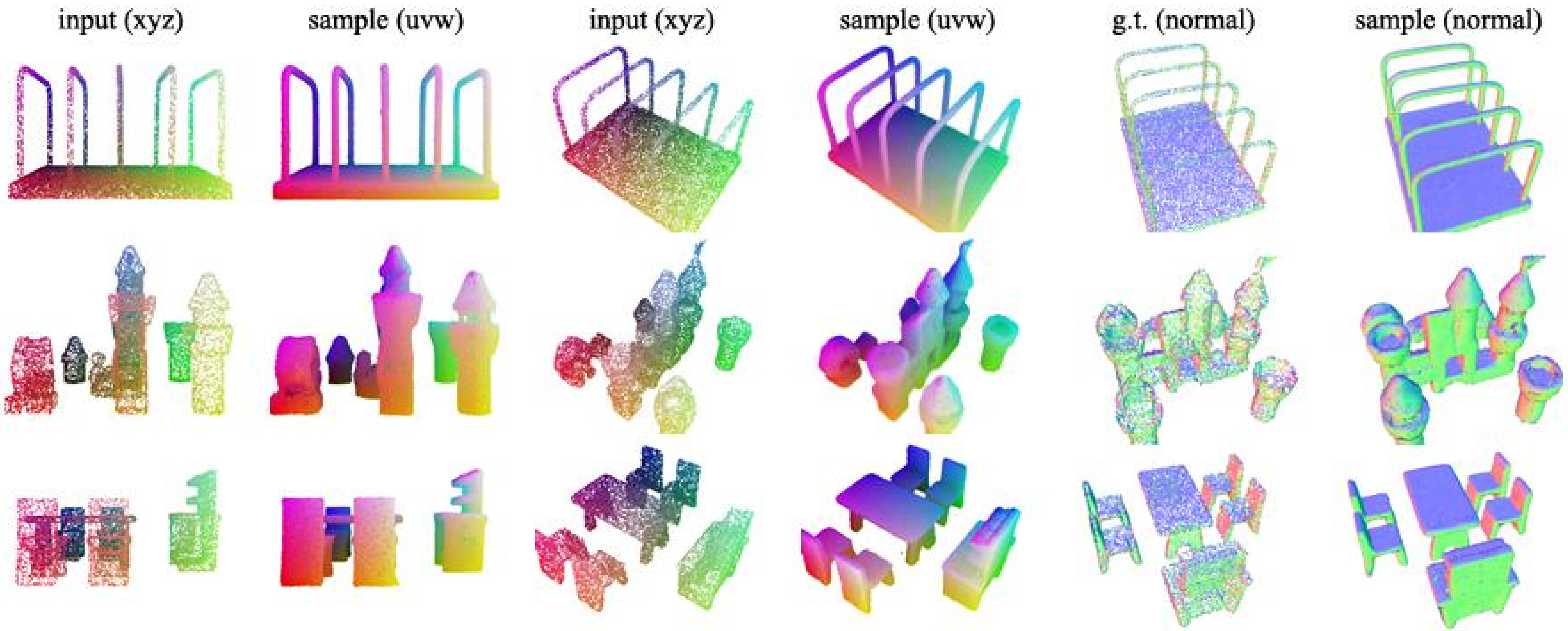
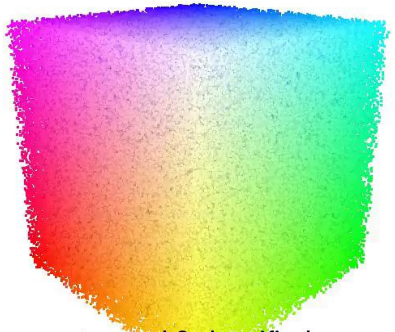


Figure 3. Reconstruction, densification, and normal estimation of unseen point clouds in GSO dataset. For each row, we are given a point cloud containing 16,384 points (xyz only), we compute ST and i.i.d. sample the resulted $p(x|s)$ for 262,144 points. Different columns render the input and the sampled point clouds from different view points. Indicated by the label in the parenthesis, we color the input points according to their xyz coordinates and the sampled points according to their initial noise's uvw coordinates and their estimated normal (last two columns). Note that we do not provide normal as input to the shape tokenizer. Mesh credits [23].

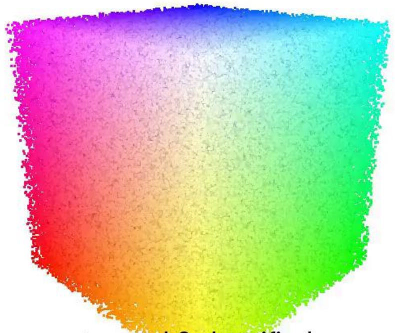
实验结果

input point cloud



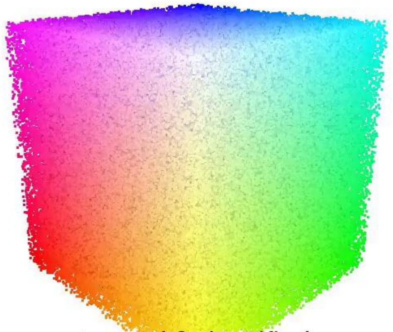
reconstructed & densified

input point cloud



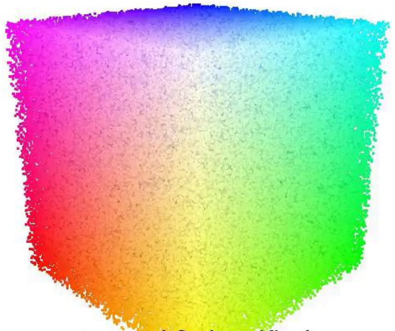
reconstructed & densified

input point cloud



reconstructed & densified

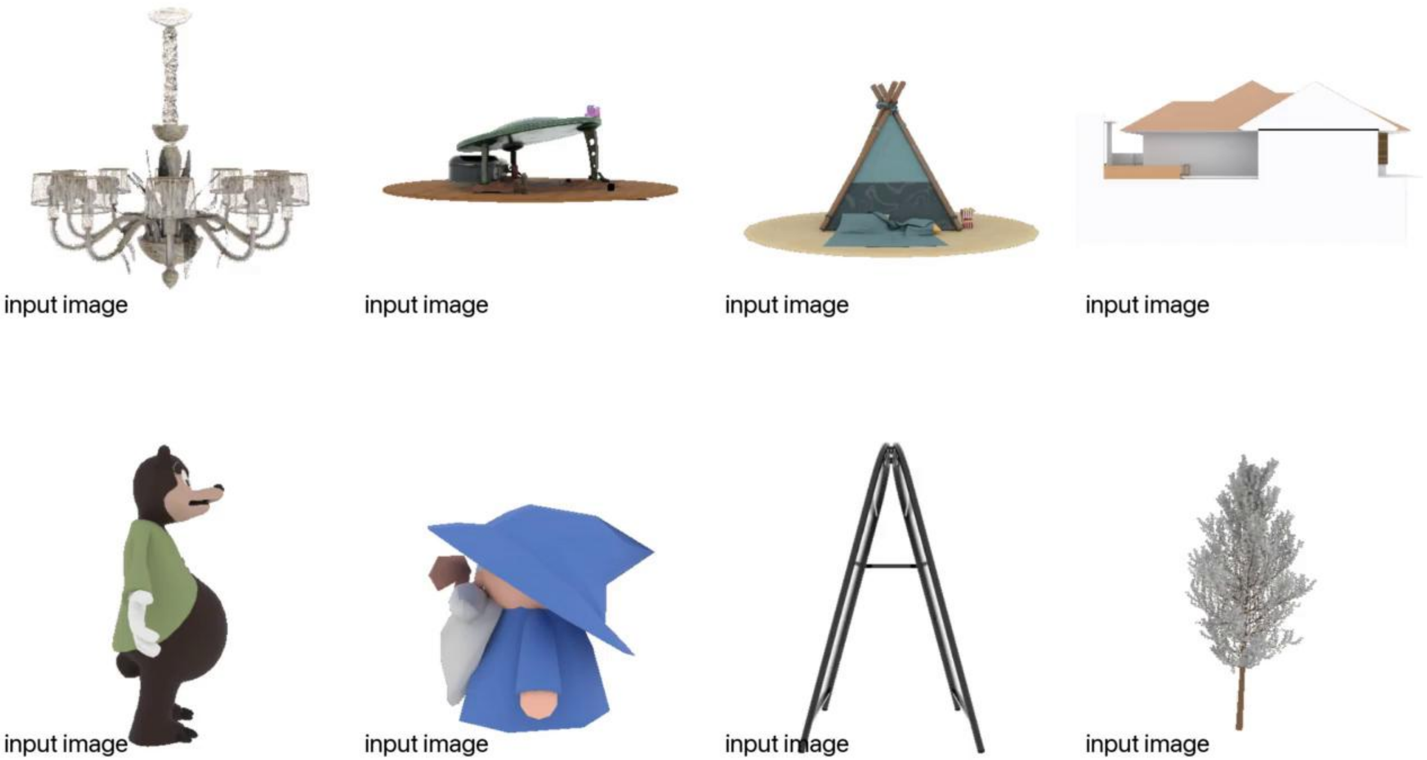
input point cloud



reconstructed & densified

应用

- 3D物体生成（single-image to 3D 或 无约束生成）
- 3D CLIP
- 神经渲染



应用

- 3D物体生成（single-image to 3D 或 无约束生成）
- 3D CLIP
- 神经渲染

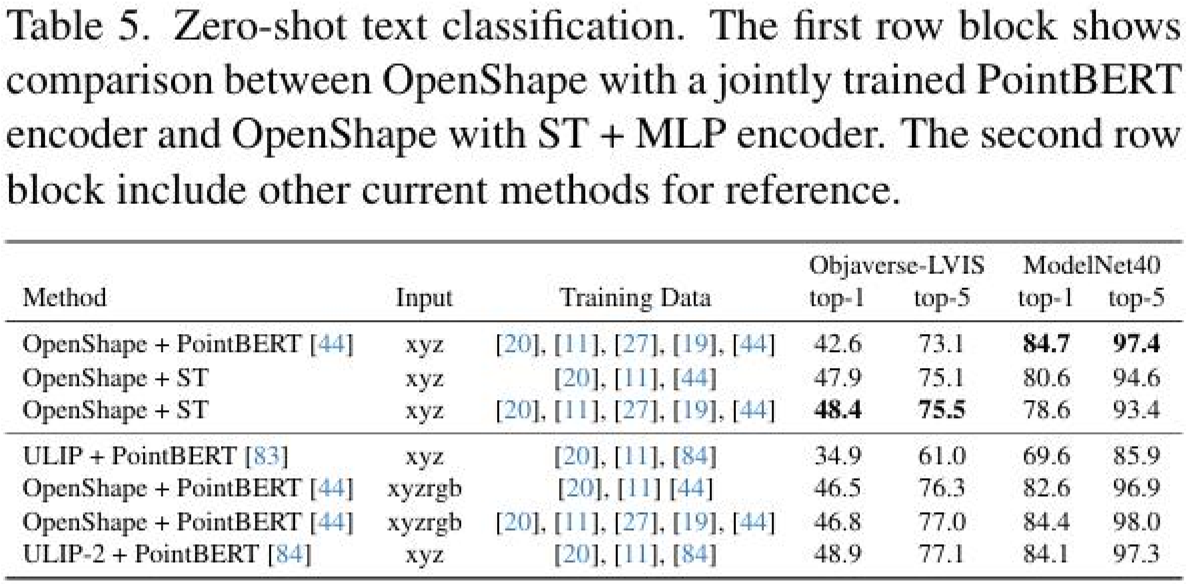


Table 5. Zero-shot text classification. The first row block shows comparison between OpenShape with a jointly trained PointBERT encoder and OpenShape with ST + MLP encoder. The second row block include other current methods for reference.

Method	Input	Training Data	Objaverse-LVIS top-1	top-5	ModelNet40 top-1	top-5
OpenShape + PointBERT [44]	xyz	[20], [11], [27], [19], [44]	42.6	73.1	**84.7**	**97.4**
OpenShape + ST	xyz	[20], [11], [44]	47.9	75.1	80.6	94.6
OpenShape + ST	xyz	[20], [11], [27], [19], [44]	**48.4**	**75.5**	78.6	93.4
ULIP + PointBERT [83]	xyz	[20], [11], [84]	34.9	61.0	69.6	85.9
OpenShape + PointBERT [44]	xyzrgb	[20], [11] [44]	46.5	76.3	82.6	96.9
OpenShape + PointBERT [44]	xyzrgb	[20], [11], [27], [19], [44]	46.8	77.0	84.4	98.0
ULIP-2 + PointBERT [84]	xyz	[20], [11], [84]	48.9	77.1	84.1	97.3

应用

- 3D物体生成（single-image to 3D 或 无约束生成）
- 3D CLIP
- 神经渲染

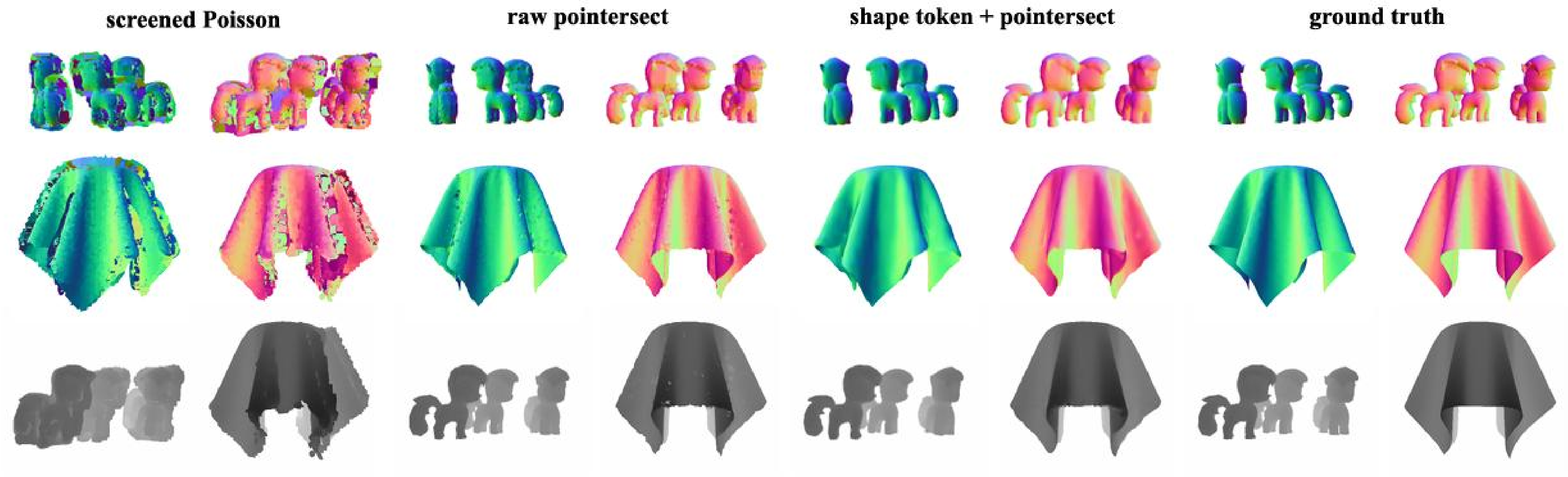


Figure 7. Given a point cloud containing 16,384 points (xyz only), camera pose and intrinsics, we process rays corresponding to each pixel individually and rasterize depth (bottom row) and normal (top two rows) map images. Mesh credits [10, 68].