3DGS：用于实时辐射场渲染的三维高斯飞溅论文复现

摘要

近些年 Neural Radiance Field 方法在多张照片或视频捕捉的场景新视角合成方面取得了革命性进展。然而，实现高视觉质量仍需要昂贵的神经网络进行训练和渲染，而最近的一些训练或渲染更快方法不可避免地在速度和质量之间做出权衡。目前没有任何方法能够实现实时显示无界完整场景（而不仅仅是孤立的对象）和 1080p 分辨率渲染。为了解决这一问题，3DGS 引入了三个关键元素，使能够在保持较短训练时间的同时实现最先进的视觉质量，并且重要的是，允许以 1080p 分辨率进行高质量的实时新视角合成。首先，从摄像机校准期间产生的稀疏点开始，3DGS 使用三维高斯来表示场景，以在场景优化中保留连续体辐射场的理想特性，同时避免在空白空间中进行不必要的计算；其次，3DGS 提出执行交替优化/密度控制的三维高斯，特别是优化各向异性协方差以实现场景的准确表示；第三，3DGS 开发了一种快速的可见性感知渲染算法，支持各向异性的投影绘制，加速训练并允许实时渲染。为了方便日后的学习研究，本文根据其开源代码进行简化重写，仅使用官方实现给的快速可微光栅化器，以求得到和官方实现相同或更好的效果。

关键词：新视角重建；3d 高斯场景表示；快速训练和渲染

1 引言

本人的研究方向是三维重建，3DGS(3D Gaussian Splatting for Real-Time Radiance Field Rendering) [3] 是今年 NeRF 方面的突破性工作，它的特点在于重建质量高的情况下还能接入传统光栅化，优化速度也快，最近也有不少文章采用 3DGS 的技术，复现这篇论文的代码对我理解三维重建和新视角生成领域相关的论文将会有很大的帮助。

2 相关工作

与 3DGS 直接相关的工作共有三个，下面一一阐述：

2.1 传统的场景重建和渲染

最早的新视图合成方法基于光场，首先密集采样，然后允许非结构化捕获。运动结构（SfM）[10] 的出现使一个全新的领域成为可能，其中一组照片可以用来合成新的观点。SfM 在相机校准过程中估计了一个稀疏点云，最初用于三维空间的简单可视化。随后的多视图立体（MVS）

[9] 多年来产生了令人印象深刻的完整三维重建算法，使几种视图合成算法得以发展。所有这些方法都将输入的图像重新投影并混合到新的视图相机中，并使用几何图形来指导这种重新投影。这些方法在许多情况下产生了极好的结果，但当 MVS 产生不存在的几何形状时，通常不能从未重建的区域或“过度重建”中完全恢复。最近的神经渲染算法极大地减少了这种伪影，并避免了在 GPU 上存储所有输入图像的巨大成本，在大多数方面都优于这些方法。

2.2 神经渲染和辐射场

神经辐射场（NeRFs）[5] 引入了重要采样和位置编码来提高质量，但使用了一个大的多层感知器，对速度产生负面影响。NeRF 的成功导致了解决质量和速度的后续方法的爆炸，通常是通过引入正则化策略；目前用于新视图合成的最先进的图像质量是 Mip-NeRF360 [1]。虽然渲染质量很优秀，但训练和渲染时间仍然非常高。最近的一些加速训练和/或渲染方法包括以下几点：1. 使用空间数据结构来存储（神经）特征，随后在体积射线行进过程中进行插值；2. 不同的编码和 MLP 容量。典型的 InstantNGP [6] 使用哈希网格和占用网格来加速计算，使用更小的 MLP 来表示密度和外观；Plenoxels [11] 使用稀疏体素网格来插值连续密度场，并且能够完全放弃神经网络。两者都依赖于球谐函数：前者直接表示方向性效应，后者直接编码其输入到颜色网络。虽然两者都提供了出色的结果，但这些方法仍然难以有效地表示空空间，这部分取决于场景/捕获类型。此外，图像质量在很大程度上受到用于加速的结构化网格的选择的限制，由于需要为给定的光线行进步骤查询许多样本，阻碍了渲染速度。3DGS 使用的非结构化、显式 gpu 友好的三维高斯算法在没有神经网络的情况下实现更快的渲染速度和更好的质量。

2.3 基于点的渲染和辐射场

基于点云的方法高效地渲染断开连接和非结构化几何样本。在其最简单的形式中，点采样渲染使用固定大小的非结构化点集进行光栅化，可以利用图形 API 的本机支持点类型或在 GPU 上进行并行软件光栅化 [8]。尽管点采样渲染忠实于底层数据，但存在孔洞、走样和严格不连续的问题。关于高质量点云渲染的开创性工作 [2] 通过使用大于像素的范围的“splatting”点基元来解决这些问题，例如圆形或椭圆形盘、椭球体或表面元素。

最近一些工作中 [7]，点云可以通过神经特征进行增强，并使用 CNN 进行渲染，实现了快速甚至实时的视角合成；然而，它们仍然依赖于多视图立体（MVS）进行初始几何生成，因此在困难情况下，如无特征/光滑区域或薄结构中，它们会继承其伪影，最明显的是过度或不足的重建。最新的方法在这个类别中不需要 MVS，使用 SH 表示方向；但它只能处理一个对象的场景，需要进行初始化。3DGS 使用 3D 高斯图来更灵活地表示场景，避免了对 MVS 几何体的需要，并通过针对投影的高斯图的基于 tile 的渲染算法实现了实时渲染。

3 本文方法

3.1 本文方法概述

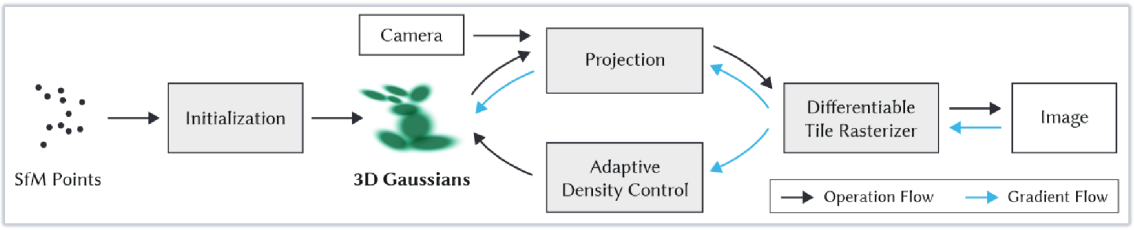

图 1. 方法示意图

上图为整个系统的示意图，系统首先对 SfM 点云进行初始化，得到 3D 高斯点云，然后借助相机外参将点投影到图像平面上（即 Splatting），接着用可微光栅化，渲染得到图像。得到渲染图像 Image 后，将其与 Ground Truth 图像比较求 loss，并沿蓝色箭头反向传播。蓝色箭头向上，更新 3D 高斯中的参数，向下送入自适应密度控制中，更新点云。

3.2 可微 3D 高斯飞溅

3.2.1 3D Gaussian

3D Gaussian 也叫三维高斯分布或三维正态分布，是统计学中一种特殊的多维正态分布，通常用来建模具有连续性随机性的现象，如图像处理、统计建模、机器学习等领域。3D Gaussian 能够涵盖空间中任意形状的椭球，包括平移、旋转。3D Gaussian 在三维空间中定义了一个概率分布，标准模型数学表示如下：

$$G_s(\mathbf{x}) = \frac{1}{(2\pi)^{3/2}\sqrt{\det(\Sigma)}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)$$

其中 $\mathrm{x} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$ 为三维坐标向量， 是椭球的中心，控制椭球的位置平移；

$$\Sigma = \begin{bmatrix} \sigma_a^2 & \mathrm{Cov}(b,a) & \mathrm{Cov}(c,a) \\ \mathrm{Cov}(a,b) & \sigma_b^2 & \mathrm{Cov}(c,b) \\ \mathrm{Cov}(a,c) & \mathrm{Cov}(b,c) & \sigma_c^2 \end{bmatrix}$$

为协方差矩阵，控制椭球在三维复现的伸缩和旋转。

论文中的 3D Gaussian 表示去掉了指数部分前面的尺度系数（因为系数不影响椭球的几何形状）和均值（因为每个点云有自己的位置），方便旋转放缩。于是 3D Gaussian 表示如下：

$$G(\mathbf{x}) = \exp\left(-\frac{1}{2}\mathbf{x}^T\Sigma^{-1}\mathbf{x}\right)$$

3.2.2 渲染

在渲染过程中一般要经过视图变换 (View Transformation) 和投影变化 (Project Transformation)，本文也是这样，先将 3DGS 经过视角变换，变换到相机坐标系，然后经过投影变换到图像中，但这存在一个问题：视角转换中 view transformation matrix 通常是仿射变换，具有仿射不变性，所以对于 3D 高斯，能够保留其分布但是对于投影 projection matrix，3D 中平行的直线会变成相交的直线，所以不能够保持不变性，作者为了保持投影变换后的高斯分布，作者做了如下方案：

对投影变换采用仿射近似：取一个点对 projection matrix 矩阵二阶泰勒展开即可得到仿射近似，更具体来说，其中的雅可比矩阵是投影矩阵在这一点对相机空间点的偏导那么协方差矩阵，其中 W 是视角变换矩阵，J 为近似的投影矩阵$\Sigma^{'} = JW\Sigma W^T J^T$

对于颜色渲染，文中通过求球谐系数 sh 从而获取对应方向上的颜色, 并且与 NeRF 相似为每个 3DGS 预测一个不透明度 ，最终整个渲染过程公式为：

$$C(p) = \sum_{i \in N} c_i f_i^{2D}(p) \prod_{j=1}^{i-1} (1 - f_j^{2D}(p))$$

其中 i 表示光线上的第 i 个 3D 高斯，那么第 i 个高斯的颜色为前面 i-1 个高斯不透明度乘以颜色乘以当前第 i 个高斯的颜色乘以对应第 i 个 3D 高斯对应 Splatting 到 2D 的高斯，在计算光线上高斯的颜色采用的和 NeRF 类似不透明度混合方式。

Splatting 是一种用于光栅化 3D 对象（如前文讨论的椭球）的技术。这些 3D 对象被映射到投影平面后得到的 2D 图形称为 splat，类似于一个点、圆、矩形或其他形状，就像雪球打在墙上留下的印记，能量从中心向外扩散并减弱。这个过程可以在 GPU 上并行处理，因为每个 Splat 之间是独立的。下图为一个 Splatting 示意图：

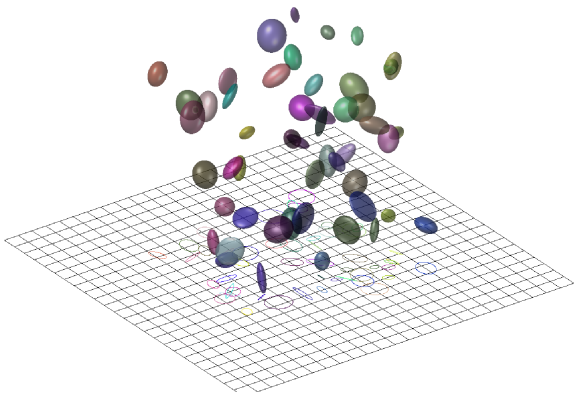

图 2. Splatting 示意图

3.3 优化与自适应密度控制

论文中一个高斯点云中存储了四个东西，分别为：

1) 点的位置（也即 3D 高斯的均值）
2) 协方差矩阵，决定高斯形状
3) 不透明度 ，用于 Splatting
4) 球谐函数 sh，拟合视角相关的外观

3.3.1 损失函数

在 3DGS 中没有使用任何卷积和神经网络，而且是利用反向传播来求解和优化上面四个东西，其损失函数为：

$$L = (1 - \lambda)L_1 \ + \ \lambda \ L_{D-SSIM}$$

比例因子设置为 0.2，SSIM 损失是图像相似度，而 D-SSMI 为 1-SSIM(X,Y)/2

3.3.2 自适应控制的高斯

3DGS 会根据梯度来控制高斯点，因为重建不充分的区域往往会有较大的梯度。当一个小尺度的几何体没有被充分重建的时候就会将原来的高斯 copy 一份，然后继续训练；当一个小尺度的几何体被一个大的高斯覆盖的时候，会将大的高斯划分成两个小的高斯。

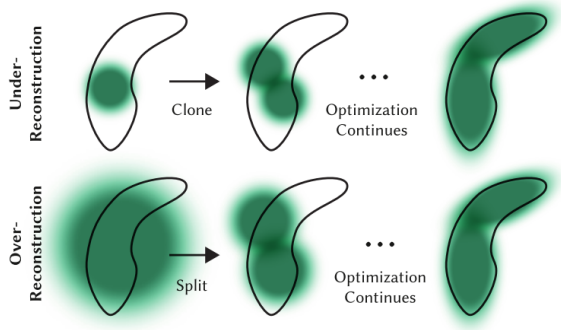

图 3. 自适应控制示意图

除此之外还会每 100 次迭代移除不透明度小于阈值的点；周期性移除较大的高斯用于避免重叠；

3.4 快速可微光栅化

为了进行快速的整体渲染和快速的排序，以允许近似的不透明度混合——包括各向异性的 Splatting——并避免对以前工作中存在的可以接收梯度的 Splatting 数量的严格限制，3DGS 用 CUDA 实现了一个 tile-based 快速可微光栅化器，具体做法如下：

1. 把整个图像划分为 16*16 个 tiles，每个 tile 视锥内挑选可视的 3D Gaussian；
2. 每个视锥内只取半透明度大于 99% 的高斯，并按深度排序；
3. 并行地在每个 tile 上 splat；有像素的不透明度达到饱和就停止对应线程；
4. 反向传播误差时按 tile 对高斯进行索引；

4 复现细节

4.1 与已有开源代码对比

本次复现，我直接使用了官方发布的快速可微光栅化器 [4]，在参考了 3DGS 的官方源码 [4] 的基础上，根据自己的学习和理解重写了 pytorch 版本的代码，代码更加简单易懂，便于学习和后续修改。

4.2 实验环境搭建

创建 conda 环境，并进行配置，python 版本为 3.9；
激活环境，安装对应的 torch 版本；安装 requirements.txt 中的包
下载 nerf_synthenic 数据集放在 dataset 目录下
运行 train.py 并配置相关参数即可。

4.3 创新点

本文的创新点在于：
我尝试使用不同损失函数进行改进，经过实验发现将原论文中 L1 损失换成:

$$\text{Smooth L1 Loss}(x, y, \beta) = \begin{cases} 0.5\left(\frac{|x-y|^2}{\beta}\right), & \text{if } |x-y| < \beta \\ |x-y| - 0.5\beta, & \text{otherwise} \end{cases}$$

smooth L1 Loss 的优点如下：(1) 真实值和预测值差别较小时（绝对值差小于 0.1），梯度也会比较小（损失函数比普通 L1 loss 在此处更圆滑）。(2) 真实值和预测值差别较大时，梯度值足够小（普通 L2 loss 在这种位置梯度值就很大，容易梯度爆炸）。

5 实验结果分析

本次复现先实验训练了原数据集 lego 模型，复现结果和改进结果如下图所示：

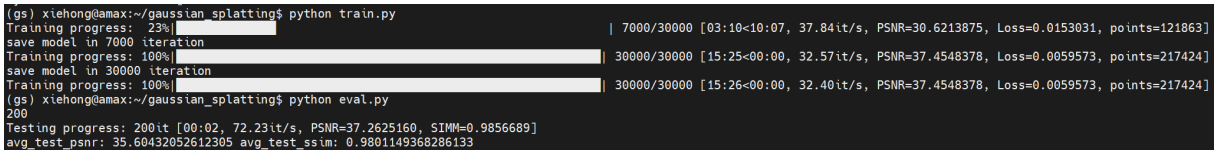

图 4. 复现运行结果

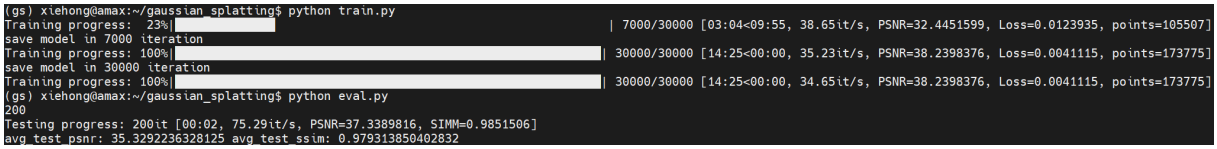

图 5. 改进结果

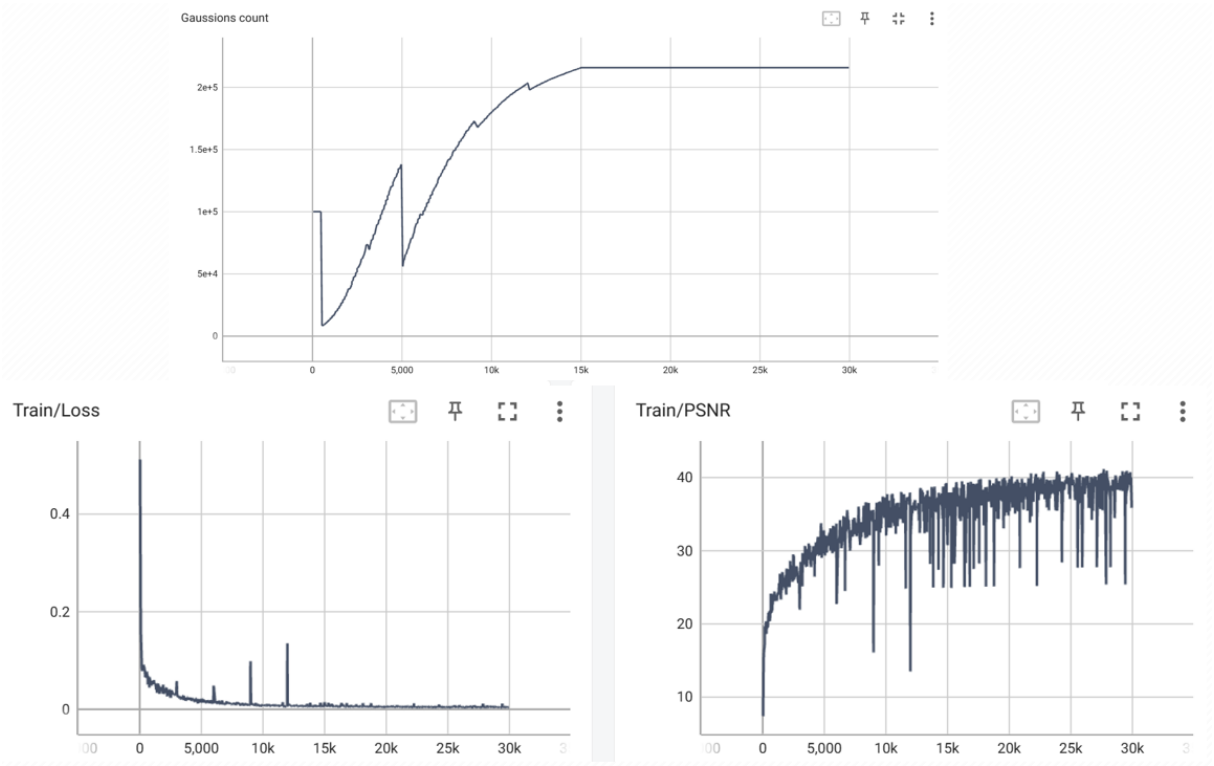

图 6. 复现代码训练过程

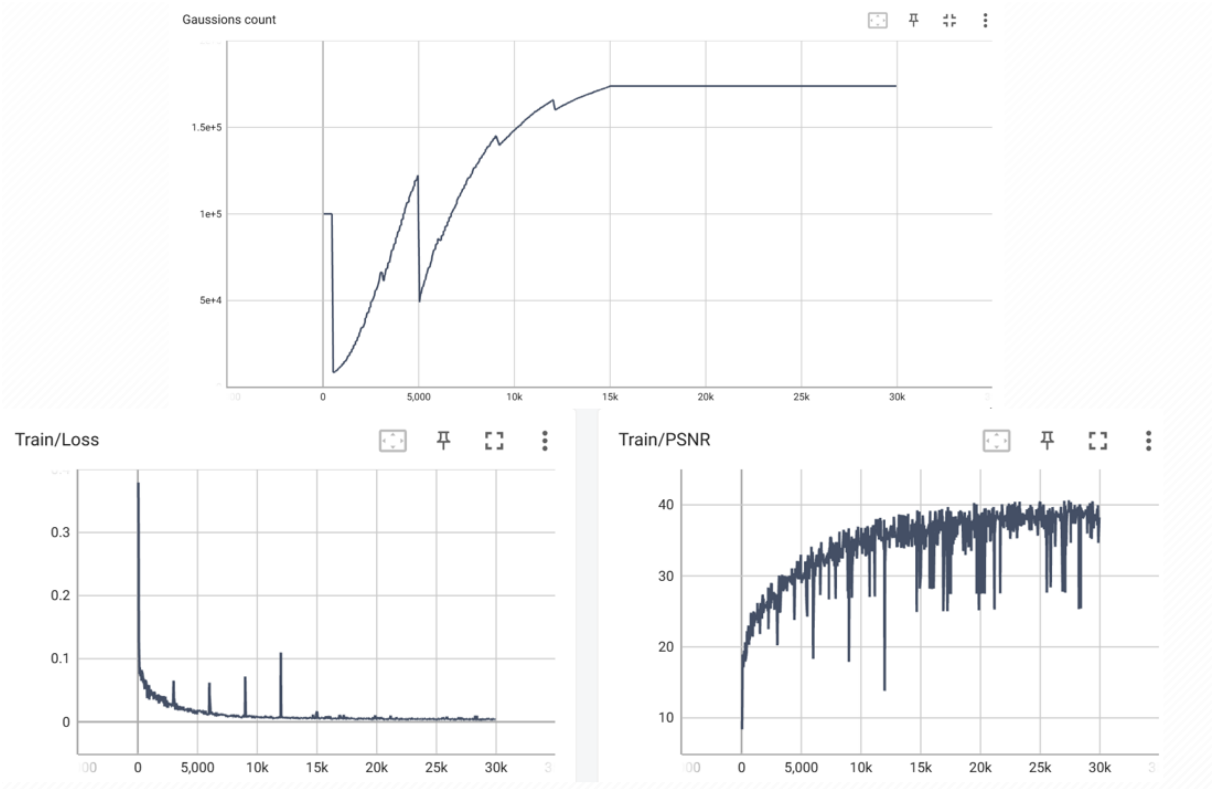

图 7. 改进后训练过程

与官方实现的效果相对比，原论文中在 lego 数据集上的 psnr 为 35.78，而我的复现效果为 35.60，略低于官方实现效果。

可以看到改进后 psnr 降低了一点点，但是可以在减少一点点 psnr 的情况下，减少不少 3D 高斯的数量，在训练时需要高斯点更少了，也就意味着训练要优化参数更少，训练时间更少。分析原因是在使用Smooth L1 Loss，在对高斯进行自适应控制时能更敏锐的找出那些需要调整的高斯点，进而对它们去除或者分裂。但高斯点数量更少，更难表示复杂的场景，所以 psnr 有所降低，但这也在误差范围内。

6 总结与展望

本文对 3DGS 进行了详细的介绍和复现，在参考官方代码和使用官方给定的快速可微光栅化器的情况下实现了和官方代码差不多的效果，并且尝试对其进行简单的改进，由于时间和能力有限未能取得很惊艳的效果，但是在这个过程中我系统前面的学习和理解了 3DGS 的原理和实现，为我今后的科研打下坚实的基础。未来准备结合 3DGS 的优点，将其应用在三维重建的细分领域，例如动态人体重建，one image to 3D 等。

参考文献

[1] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5460–5469, 2021.

[2] Mario Botsch, Alexander Sorkine-Hornung, Matthias Zwicker, and Leif Kobbelt. High-quality surface splatting on today's gpus. *Proceedings Eurographics/IEEE VGTC Symposium Point-Based Graphics, 2005.*, pages 17–141, 2005.

[3] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkuehler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (TOG)*, 42:1 – 14, 2023.

[4] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023.

[5] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM*, 65:99–106, 2020.

[6] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (TOG)*, 41:1 – 15, 2022.

[7] Darius Rückert, Linus Franke, and Marc Stamminger. Adop. *ACM Transactions on Graphics (TOG)*, 41:1 – 14, 2021.

[8] Markus Schütz, Bernhard Kerbl, and Michael Wimmer. Software rasterization of 2 billion points in real time. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 5:1 – 17, 2022.

[9] Steven M. Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, 1:519–528, 2006.

[10] Noah Snavely. Photo tourism : Exploring image collections in 3d. *ACM Transactions on Graphics*, 2006.

[11] Alex Yu, Sara Fridovich-Keil, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5491–5500, 2021.