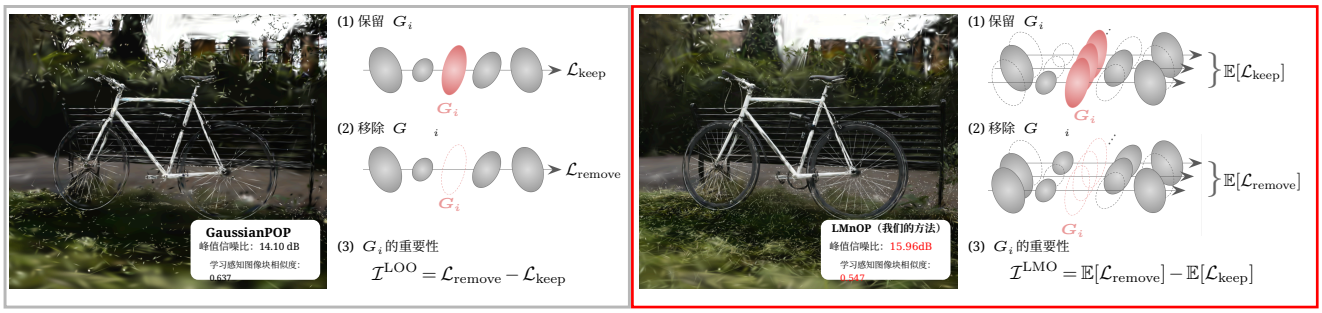
arXiv:2611.30102v1 [cs.CV] 23 Nov 2026

LMnOP：基于留多法重要性的3D高斯泼溅剪枝

Sunjae Jeong[1,2] Jaeseung Kim[1] Hyoseok Hwang[2], *[1]
MOBILTECH CO.,LTD. [2] 庆熙大学{sunj, jason.kim}
@mobiltech.io hyoseok@khu.ac.kr



(a) 留一法 ***(b)*** 留多法（我们的方法）

图1. **剪枝掉99%的高斯**。 Bicycle场景，未进行微调，峰值信噪比和学习感知图像块相似度为其测试视图的平均值。 **(a)** 图中显示的是 GaussianPOP 的结果 [1], ，它使用 留一法重要性。 留一法 根据移除某个高斯且保留其他所有高斯时 G_i 损失增加的程度为其评分 G_i 。当多个高斯覆盖同一区域时，它们各自单独看来都像是冗余的，因此会被一并移除，留下空洞。 **(b)** 因此，我们提出了 留多法，它根据随机移除其他高斯时的平均损失增加为 G_i评分。 留多法 会保留那些 留一法会一并移除的高斯。

摘要

*3D*高斯泼溅能够实时渲染出照片级真实感视图。然而，单个场景通常需要数百万个高斯，因此这些模型的存储和渲染成本高昂。剪枝通过保留高斯的子集来降低这一成本。一种常用的评分是留一法重要性，即单独移除某个高斯时损失的增加量。在较高的剪枝率下，许多高斯会被同时移除，因此那些单独来看具有冗余性、但共同存在时不可或缺的高斯会被丢失。我们提出了***LMnOP***（*Leave-Many-Out Pruning*，留多法剪枝），用于估计在移除其他高斯的情况下每个高斯的重要性。这种留多法重要性是指：在其他高斯也被随机移除时，移除某个高斯所导致的损失增加的均值。我们以闭式形式计算随机移除下每个像素颜色的精确均值和方差，并据此近似重要性，而无需采样移除模式。我们利用每个高斯的重要性来细化其保留概率，并在最后一次性进行剪枝。

重要性，并在最后一次性进行剪枝。在不进行微调的情况下，***LMnOP*** 在三个数据集、*90%*和*99%*的移除率下，在*18*项指标中的*16*项上取得了最佳结果。在全部*13*个场景中，相较于*GaussianPOP*，它都提升了结构相似性指数和学习感知图像块相似度。

1. 引言

3D高斯泼溅（3DGS） [2]使用3D高斯表示场景，并实时渲染出照片级真实感视图。凭借质量与速度兼具的优势，3DGS迅速成为表示3D场景的常用选择。然而，单个场景往往需要数百万个高斯，因此模型的存储和渲染成本高昂，尤其是
在内存有限的设备上。人们探索了多种降低这一成本的途径。一种方法通过量化和编码来减少每个高斯所需的比特数 [3–5]。这能降低存储开销，但高斯的数量以及由此产生的渲染成本都保持不变。另一种方法是在训练过程中生成更少的高斯 [6]，但这需要重新训练模型。剪枝则

*通讯作者。

只保留训练好的模型中实际需要的高斯。它能同时减少存储开销和渲染成本，并且可以与压缩结合使用。移除的高斯越多，节省就越大，而保留哪些高斯也就越重要。

为了决定保留哪些高斯，大多数剪枝方法都会为每个高斯分配一个重要性分数，并保留分数最高的高斯。这些分数形式各异，例如累积混合权重 [6, 7], 对参数变化的敏感性 [8], 以及移除单个高斯所导致的图像误差 [1]。尽管形式不同，这些分数都是在模型其余部分保持不变的情况下计算的。最直接体现这一思想的方式，是单独移除一个高斯时损失的增加。我们将这一量称为留一法重要性。在 Gaussian-POP [1], 中，对于平方误差，该量可以精确计算。当只移除少量高斯时，这种分数很准确，因为模型其余部分几乎不会发生变化。

然而，当必须移除的高斯数量多得多时，通常会一次移除多个，而不是逐个移除。考虑一个由多个高斯共同覆盖的区域。如果移除其中任意一个，其余高斯仍会填满该区域，因此每个高斯的留一法重要性都很低。如图1(a)所示，移除所有这些高斯后就会留下空洞。这些高斯单独来看是冗余的，但共同作用时却不可或缺；然而，它们的留一法重要性却与真正不必要的高斯一样低。迭代剪枝通过分多个阶段移除高斯，并在每个阶段后重新评估剩余高斯的分数，从而缓解这一问题 [1]。

我们研究了这样一个问题：能否在移除任何高斯之前，就将移除其他高斯所产生的影响纳入重要性本身。评估某个高斯时，应将其置于剪枝实际造成的情境中，即许多其他高斯也会被一并移除。遵循这一原则，我们提出了*LMnOP*，通过其留多法 (LMO) 重要性为每个高斯评分（图 1（b））。该重要性表示：在其他高斯也按照各自的保留概率随机移除的情况下，移除该高斯所导致的平均损失增加。当所有保留概率均为 1 时，LMO 重要性便退化为留一法重要性。保留概率应反映哪些高斯将会保留，但这在事先并不可知。因此，我们在多个步骤中利用重要性对这些概率进行优化，使每个高斯都能在接近最终选择的情境下接受评估。在这些步骤中不会移除任何高斯，因此选择结果仍可发生变化。最后执行一次剪枝，仅保留具有最高重要性的高斯。

计算 LMO重要性是主要难点。它是对所有移除模式取平均，而渲染每种模式的代价高得难以承受。我们提出 μ-σ 渲染，可用闭式形式计算随机移除下每个像素颜色的精确均值和方差。根据这两项统计量，我们无需采样移除模式即可计算所有高斯的重要性。对于平方误差，该重要性是精确的；对于我们使用的损失，则是近似的。

我们的贡献如下：

- 我们定义了 LMO重要性，用于评估每个高斯，同时随机移除其他高斯。我们根据该重要性优化保留概率，并在最后一次性完成剪枝。
- 我们提出了 μ-σ 渲染方法，以闭式形式计算随机移除情况下每个像素颜色的精确均值和方差。借助这些精确统计量，我们无需对移除模式进行采样，即可近似计算所有高斯的重要性。
- 无需微调，*LMnOP* 在三个数据集、90%和99%的移除率下，在18项指标中的16项上取得了最佳结果。我们还表明，它能够保留剪枝后重要性上升的高斯。

2. 相关工作

Compact 3DGS。 3DGS 模型 [2]可以通过两种互补的方式实现紧凑化。一种方式是减少表示每个高斯所需的比特数，另一种方式是减少高斯的数量。Compact3DGS [3], EAGLES [4], 和 MesonGS [5] 将紧凑的属性表示与缩小模型规模的机制相结合。相比之下，Mini-Splatting [6]在训练过程中直接生成更少的高斯。对训练完成的模型进行剪枝是减少高斯数量最直接的方法，也可以与属性压缩结合使用。然而，其结果完全取决于哪些高斯被判定为重要。

基于重要性的剪枝。大多数剪枝方法都会为每个高斯分配一个分数，并保留分数最高的高斯。LightGaussian [7] 和 Mini-Splatting 衡量每个高斯对渲染的贡献。PUP 3D-GS [8]通过二阶近似损失的敏感度。GaussianPOP [1] 会解析计算移除单个高斯造成的误差。Trimming [9] 使用梯度信息，而 R AP [10] 则根据高斯属性和邻域统计量预测重要性。还有一些方法会学习掩码，而非计算分数 [3, 11],，但它们必须重新优化模型。例如，MaskGaussian 会在优化过程中采样概率掩码，并为被掩码的高斯保留梯度。上述分数的形式各不相同，但都是在模型其余部分仍然存在时计算得出的。因此，它们反映的是单个高斯的贡献，而不是同时移除大量高斯时会发生什么。

交互感知剪枝。 高斯被同时移除时会发生什么，取决于它们如何相互作用，因为移除一个高斯会改变其后方每个高斯的透射率。

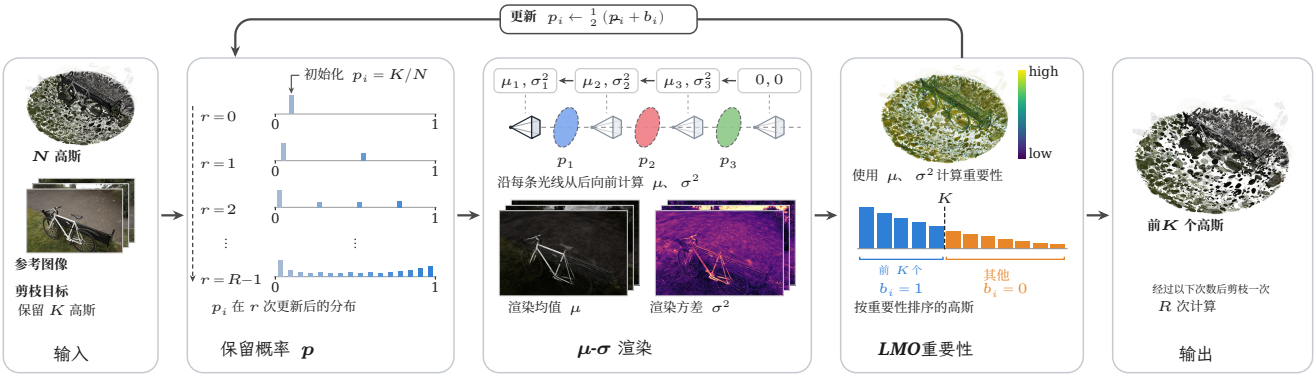


图 2. **LMnOP 概述*LMnOP***。每个高斯的初始保留概率均为 K/N，直方图显示了经过 r 次更新后的保留概率。基于当前保留概率进行 μ-σ 渲染，可得到每个像素颜色的均值和方差，我们据此计算每个高斯的 LMO 重要性。然后，我们将每个保留概率向 b_i 移动一半，并再次计算重要性。这里，b_i 对于重要性最高的 K 个高斯为 1，对于其他高斯为 0。经过 R 次重要性计算后，我们保留这些 K 个高斯，并在最后一步中剪枝其余高斯。

在网络剪枝中也存在同样的问题：最优脑外科 [12]及其组合扩展 [13]都会考虑权重之间的交互。在 3 DGS 中，以往工作通过三种方式处理这种交互。REFINE [14] 假设高斯彼此独立，在不进行渲染的情况下估算重要性。核心集的渲染保证 [15]依赖于有效性和对数透射率稳定性的假设。GaussianPOP 会在多个剪枝阶段重复评分，使后续评分反映先前的移除操作。同期工作 SplatQ [16], 则对空间块内移除高斯对的影响进行建模，并求解二次选择问题。因此，交互要么被忽略，要么只在不同剪枝阶段之间加以捕捉，要么仅对块内的高斯对进行建模。相较之下，我们根据每个高斯的 LMO 重要性进行评分。该重要性通过对所有其他高斯的随机移除取平均得到。因此，在移除任何高斯之前，每条光线上所有高斯之间的交互就已纳入重要性本身。

3. 预备知识

3.1. 3DGS 渲染

3DGS [2] 使用 N 高斯表示场景，每个高斯都具有位置、协方差、不透明度和视角相关颜色。为了渲染一个像素，将覆盖该像素的高斯按从前到后的顺序排列，并以 $k = 1,\ldots,n$ 编号。第 k 个高斯的颜色为 c_k，alpha 值为 α_k，即其不透明度乘以该像素处投影的二维高斯值。像素颜色随后为

$$C = \sum_{k=1}^{n} \alpha_k c_k T_k, \quad \text{with} \quad T_k = \prod_{j<k}(1-\alpha_j), \qquad (1)$$

其中，透射率 T_k 是到达第 k 个高斯的光线所占的比例。由于一个高斯在不同光线上会出现在不同深度处，我们用 i 表示模型中的一个高斯，用 k 表示其在当前光线上的位置。我们针对一个颜色通道写出所有方程，并假设背景为黑色。

3.2. 剪枝目标

剪枝会保留 K 个 N 高斯，同时不改变其属性。我们用二值掩码 $m = (m_1, \ldots, m_N)$表示保留哪些高斯，其中 $m_i = 1$保留高斯 i，而 $m_i = 0$ 则将其移除。在式（1）中，这相当于将每个 α_k 替换为 $m_k\alpha_k$，因此被移除的高斯不会产生任何贡献。为衡量剪枝造成的损害，我们将使用 m 渲染的图像与参考图像进行比较。参考图像是原始模型从某个训练视角渲染得到的结果，C^{ref} 则是其在某个像素处的颜色。在整个剪枝过程中，参考图像保持不变。剪枝损失 $\mathcal{L}(m)$通过 L1 项和 D-SSIM 项，对所有像素和视图上的比较结果取平均。剪枝问题定义为

$$\min_{m\in\{0,1\}^N} \mathcal{L}(m) \quad \text{with} \quad \textstyle\sum_i m_i = K. \qquad (2)$$

可供直接搜索的掩码数量实在太多，因此大多数方法会为每个高斯评分，并保留得分最高的 K个高斯。一种常见的评分是留一法重要性，即仅从原始模型中移除高斯i 时损失的增加量。

4. 方法

4.1. 留多法重要性

留一法重要性是在保留其他所有高斯的情况下进行衡量的。相反，我们会在

其他高斯也被随机移除的情况下评估每个高斯。我们将每个 m_i 视为一个独立的随机变量，其取值为 1 的概率是 p_i。我们称 p_i 为高斯 i的保留概率。高斯i 的留多法 (LMO) 重要性为

$$\mathcal{I}_i(p) = \mathbb{E}[\mathcal{L} \mid m_i = 0] - \mathbb{E}[\mathcal{L} \mid m_i = 1], \tag{3}$$

其中，期望是针对其他高斯计算的。较大的值意味着保留高斯 i 会降低期望损失。当所有其他高斯都保留时，LMO重要性等于留一法重要性。最简单的保留概率选择是为每个高斯设为 K/N，但重要性高的高斯比其他高斯更有可能被保留。

4.2. 重要性感知的保留概率

重要性高的高斯应获得较高的保留概率，但重要性本身取决于保留概率。我们通过一种更新方式解决这种循环依赖，该方式基于平均保留 K 个高斯时的期望损失。LMO重要性是期望损失相对于 p_i的负导数。因此，在一阶近似下，当最高重要性的 K 个高斯的保留概率为 1、其余高斯的保留概率为 0 时，该期望损失最低。令 b_i 表示高斯 i的这一数值。随着保留概率的变化，该估计值也会改变，因此我们只向该估计值移动一半：

$$p_i \leftarrow \tfrac{1}{2}\,(p_i + b_i). \tag{4}$$

此次更新会使保留概率之和保持为 K。从 K/N （即每个高斯的初始值）开始，我们交替进行重要性计算和应用公式（4）。我们总共计算重要性 R 次，中间进行 $R-1$ 次更新（图2）。每次更新都会使保留概率从 K/N 朝着0 或1移动。这些步骤中不会移除任何高斯，因此选择结果仍可能改变。最后一次计算完成后，我们保留 LMO 重要性最高的 K 个高斯，并在最后一步中剪枝其余部分。随机掩码仅用于计算重要性。

4.3. μ-σ 渲染

接下来需要计算 LMO 重要性。直接计算公式（3）中的期望，需要渲染所有可能的掩码。然而，对于平方误差，像素处的期望损失仅取决于其颜色的均值 μ和方差 σ^2：

$$\mathbb{E}\big[(C - C^{\text{ref}})^2\big] = (\mu - C^{\text{ref}})^2 + \sigma^2. \tag{5}$$

如图所示，我们沿每条光线从后向前计算这两个统计量。2 设 μ_k 和 σ_k^2 分别表示从第 k个高斯到最后一个高斯混合得到的颜色的均值和方差。保留第 k个高斯而非将其移除会使均值变化 $\delta_k = \alpha_k(c_k - \mu_{k+1})$。我们用 $\gamma_k = 1 - p_k\alpha_k(2 - \alpha_k)$ 表示通过该高斯的透射率平方的期望。于是，这两个统计量可表示为

$$\begin{aligned} \mu_k &= \mu_{k+1} + p_k\delta_k, \\ \sigma_k^2 &= \gamma_k\sigma_{k+1}^2 + p_k(1 - p_k)\delta_k^2. \end{aligned} \tag{6}$$

从最后一个高斯之后的零值开始递归，最终得到像素的均值和方差；对于独立掩码，这两者都是精确的。我们将这一计算称为 μ-σ渲染。其均值是 α-混合，其中每个 alpha 都乘以其保留概率。其方差衡量同时移除多个高斯时像素变化的幅度。

4.4. 评分损失

只有对于平方误差，均值和方差才能精确给出期望损失。实际中，我们遵循 3DGS 的训练损失 [2]，将 L1 项和 D-SSIM 项按权重 $1 - \lambda$ 和 λ 组合起来。我们将这种组合称为评分损失。对于 L1 项，我们按如下方式度量每个像素：

$$e = \sqrt{(\mu - C^{\text{ref}})^2 + \beta\sigma^2 + \varepsilon}, \tag{7}$$

其中 ε 是一个较小的常数，因此当方差为零时，e 是平滑后的绝对误差。我们对颜色通道、像素和视图上的 e取平均，得到 $\mathcal{L}_1$，而 β 决定方差所占的权重。D-SSIM 项 $\mathcal{L}_{\text{D-SSIM}}$ 在均值图像与参考图像之间计算。当每个保留概率都为 0 或 1 时，方差为零，评分损失等于式（2）中的剪枝损失，但会受到 ε平滑处理的影响。对于其他保留概率，它是期望损失的替代量。

我们将评分损失的 LMO重要性计算为

$$\mathcal{I}_i(p) \approx (1 - \lambda)\,\Delta_i\mathcal{L}_1 + \lambda\,\Delta_i\mathcal{L}_{\text{D-SSIM}}, \tag{8}$$

其中 Δ_i 表示当高斯 i 被移除而非保留时损失的变化，同时固定其他保留概率。高斯 i 被保留或移除时的像素统计量，可通过一次从前到后的遍历和一次从后到前的遍历以闭式形式得到，而无需分别渲染每个高斯。这使得 $\Delta_i\mathcal{L}_1$ 能够精确计算。对于 D-SSIM 项，我们在均值图像附近将损失展开至二阶，二阶项由 $p_i - \frac{1}{2}$ 加权。我们使用这一近似重要性来计算公式（4）中的更新。

5. 实验

5.1. 设置

数据集和指标。我们评估了 13 个场景：9 个来自 Mip-NeRF 360 [17], 2 ，1 个来自 Deep Blending [18], , 2 个来自 Tanks & Temples [19]。每个场景的原始模型都是在标准数据集分辨率下训练的 3DGS 模型 [2] 。我们测量了峰值信噪比、结构相似性指数 [20], 和学习感知图像块相似度 [21]

表1。**与基线方法的比较**。在三个数据集上移除 90% 和 99% 的高斯，且不进行微调。所有方法都将同一个原始模型剪枝至相同数量的高斯。本文中，金色、银色和铜色分别标示某列四舍五入前的三个最佳值，并列值共享排名。

	移除 90%									移除 99%								
	Mip-NeRF360			Deep Blending			Tanks & Temples			Mip-NeRF360			Deep Blending			Tanks & Temples		
方法	峰值信噪比↑	结构相似性指数↑	学习感知图像块相似度↓	峰值信噪比↑	结构相似性指数↑	学习感知图像块相似度↓	峰值信噪比↑	结构相似性指数↑	学习感知图像块相似度↓	峰值信噪比↑	结构相似性指数↑	学习感知图像块相似度↓	峰值信噪比↑	结构相似性指数↑	学习感知图像块相似度↓	峰值信噪比↑	结构相似性指数↑	学习感知图像块相似度↓
原始（0%）	27.35	0.814	0.257	29.52	0.903	0.311	23.39	0.842	0.219	27.35	0.814	0.257	29.52	0.903	0.311	23.39	0.842	0.219
Uniform	12.52	0.271	0.612	12.61	0.445	0.628	9.91	0.279	0.599	8.87	0.061	0.716	7.48	0.104	0.752	5.77	0.057	0.708
不透明度	15.36	0.378	0.561	17.91	0.621	0.539	13.91	0.452	0.512	9.76	0.125	0.701	7.93	0.135	0.745	7.00	0.141	0.691
EAGLES* [4]	**20.29**	**0.651**	**0.394**	25.04	**0.835**	**0.400**	17.88	**0.689**	**0.355**	14.20	**0.413**	**0.621**	16.61	0.692	0.577	12.30	**0.458**	**0.610**
MesonGS* [5]	19.86	0.630	**0.408**	24.88	0.832	0.404	18.07	0.678	0.368	14.26	**0.409**	0.634	16.96	**0.700**	0.580	**12.47**	**0.455**	0.619
Trimming* [9]	15.55	0.422	0.532	17.68	0.655	0.509	14.08	0.484	0.483	9.67	0.134	0.696	8.00	0.149	0.720	6.82	0.145	0.674
LightGaussian* [7]	19.16	0.578	0.450	24.34	0.817	0.425	17.66	0.649	0.394	**14.33**	0.393	0.646	**18.22**	**0.713**	0.584	12.41	0.447	0.622
PUP 3D-GS* [8]	17.05	0.578	0.442	22.55	0.798	0.432	13.55	0.601	0.414	11.18	0.256	0.653	11.31	0.479	0.643	8.91	0.320	0.631
GaussianPOP† [1]	**22.55**	**0.664**	0.409	**27.69**	**0.854**	**0.388**	**20.81**	**0.718**	**0.355**	**15.38**	0.376	**0.628**	**19.86**	0.687	**0.563**	**14.00**	0.443	**0.586**
RAP [10]	19.89	0.568	0.468	**25.40**	0.823	0.432	**18.33**	0.641	0.411	13.88	0.347	0.660	18.13	0.690	**0.576**	12.24	0.408	0.630
核心集* [15]	20.22	0.646	0.412	23.89	0.821	0.419	17.65	0.658	0.389	14.32	0.403	0.634	16.88	0.681	0.579	12.36	0.444	0.619
LMnOP（我们的方法）	**23.00**	**0.715**	**0.373**	**27.22**	**0.869**	**0.371**	**20.15**	**0.736**	**0.332**	**16.53**	**0.467**	**0.594**	**19.97**	**0.732**	**0.551**	**14.63**	**0.512**	**0.550**

†我们的实现。 *仅重要性或选择组件 y.

在测试视图上计算这些指标，并对每个数据集中的场景进行等权平均。我们使用 VGG 网络计算图像的学习感知图像块相似度，图像缩放至 $[-1, 1]$。

基线方法。 所有方法都将同一个原始模型剪枝至相同数量的高斯，且之后均未进行微调。若干基线方法属于更大的流程，这些流程还会重新训练或压缩模型。对于

对于这些方法，我们仅采用其重要性或选择组件。Uniform 会保留 K 随机选取的高斯，而不透明度方法会保留 K 不透明度最高的高斯。对于 GaussianPOP， [1], 我们按照其论文中的做法，使用了自己的实现，并设置了八个剪枝阶段。

实现细节。*LMnOP*计算重要性 $R = 8$次，同时保留所有高斯，然后在最后一步对每个模型进行剪枝。我们设置了 $\lambda = 0.2$、$\beta = 0.5$ 和 $\varepsilon = 10^{-6}$。重要性在所有训练视图上累积，并以原始模型的渲染结果作为参考图像。所有场景和剪枝比例均采用相同设置。所有实验均在 NVIDIA RTX A6000 GPU 上运行。

5.2. 与基线方法的比较

我们比较了*LMnOP* 与移除率为 90% 和 99% 时的基线方法。剪枝后未对任何模型进行微调，因此结果仅反映每种方法保留了哪些高斯。

定量结果。 如表 1 所示，*LMnOP* 在 18 列中有 16 列取得了最佳结果。GaussianPOP 和 EAGLES [4] 在大多数列中紧随其后。两个例外是两个数据集在移除 90% 后的峰值信噪比，此时 GaussianPOP 的结果更高。在结构相似性指数和学习感知图像块相似度方面，*LMnOP* 在两种剪枝比例下的全部 13 个场景中均优于 GaussianPOP。移除 99% 后，它在每个场景的这两项指标上均优于所有其他方法。

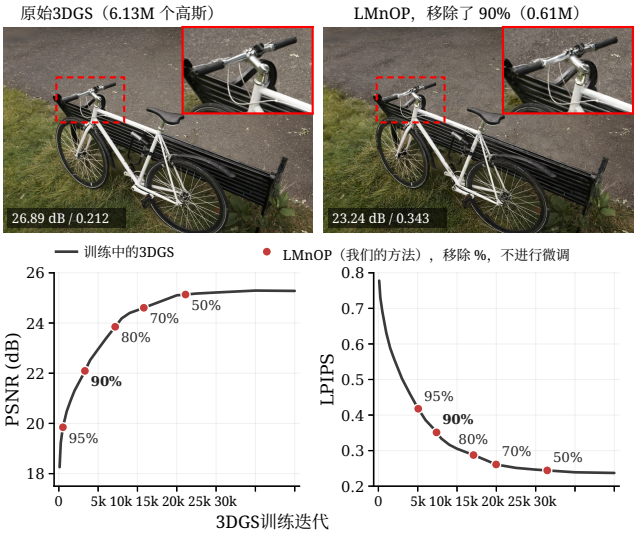
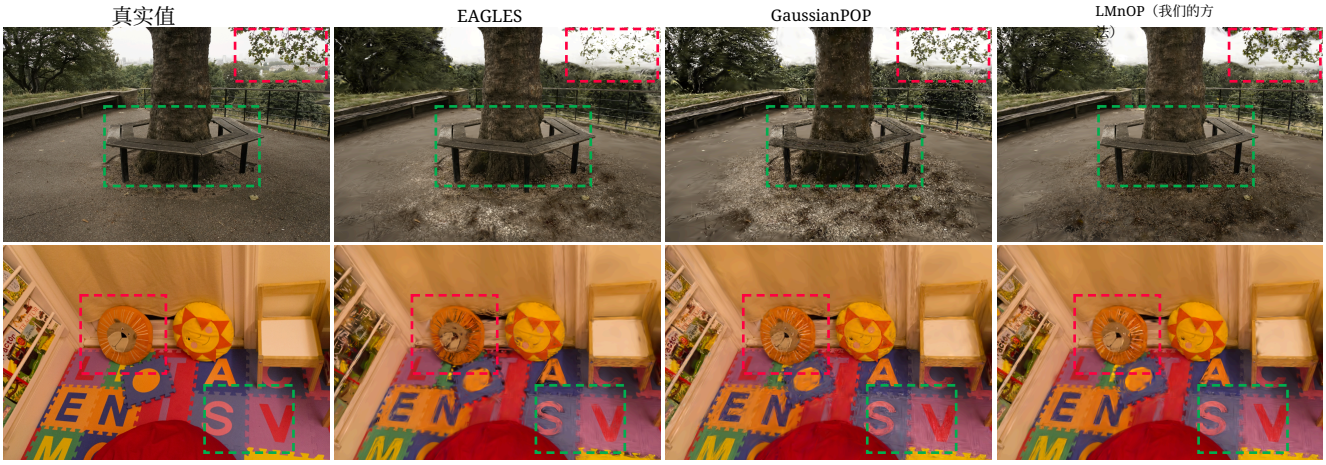


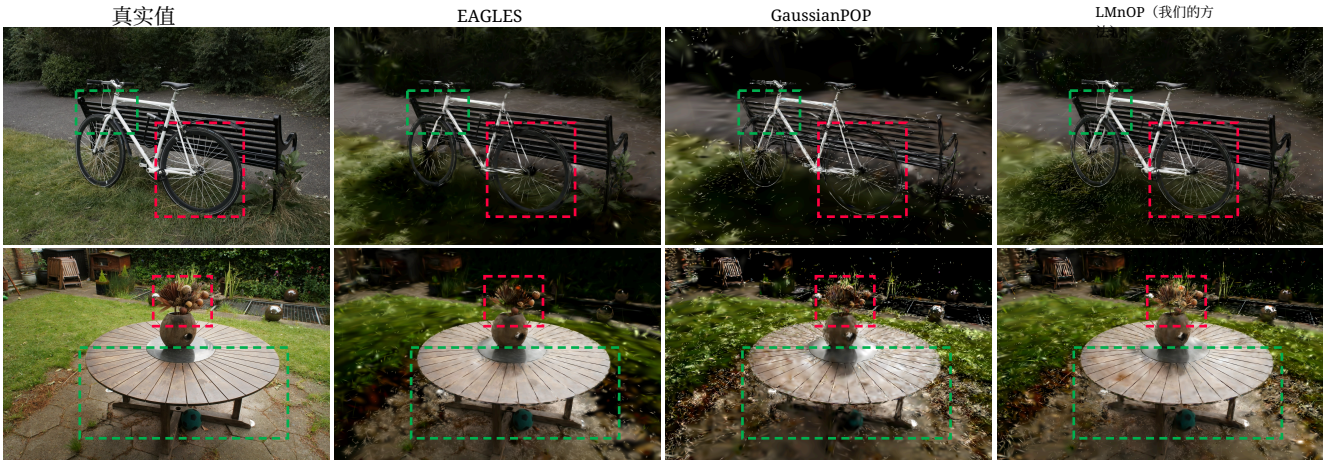
图 3. **3DGS 训练曲线上剪枝模型的质量**。Bicycle，未经微调。**上图**：原始模型和*LMnOP* 在移除率为 90% 时的结果，并标注视图的峰值信噪比 / 学习感知图像块相似度。红框中的放大区域显示剪枝模型中有轻微的白色伪影。**下图**：训练过程中 3DGS 的测试峰值信噪比和学习感知图像块相似度。每个标记都将剪枝模型放置在 3DGS 达到相同数值的位置。标记表示质量，而非训练时间。

仅凭这些指标无法看出剪枝模型与原始模型有多接近。因此，我们将剪枝模型与 3DGS 的训练曲线进行了比较（图 3）。在 Bicycle 场景中，移除 90% 高斯后的模型，其学习感知图像块相似度与 3DGS 训练约 7.4k 次（总计 30k 次）迭代后达到的水平相当；其峰值信噪比则与 3DGS 训练约 3.3k 次迭代后达到的水平相当。在示例视图中，该模型的渲染结果保留了原始模型的整体外观，尽管它仅保留了十分之一的高斯，且未经微调。

定性结果。 在图4中，我们将*LMnOP* 与 EAGLES 和 GaussianPOP 这两种最强的基线方法进行比较，比较范围为



（a）移除了 90%



（b）移除了 99%

图 4. **定性比较**。（a）移除 90% 后的 Treehill 和 Playroom，以及（b）移除 99% 后的 Bicycle 和 Garden，均未进行微调。红色方框标出高频区域，绿色方框标出低频区域。

表 1。红色方框标出具有精细细节的高频区域，绿色方框标出具有光滑表面的低频区域。两种基线方法在相反的区域表现不佳。例如，在图4(a) 顶行的 Treehill 场景中，EAGLES 对长椅的渲染效果很好，但对树叶的效果不佳。GaussianPOP 对树叶的渲染效果很好，但在长椅上产生了伪影。其他场景也呈现出相同的模式。在移除率为 99 % 时，GaussianPOP 对 Garden 场景中的桌子渲染不佳（图4(b) 底行）。*LMnOP* 在全部四个场景中，对高频和低频区域的渲染效果都很好。

5.3. 剪枝前后的重要性

上面的比较表明，*LMnOP* 选出的子集更优，但没有说明原因。我们的前提是，移除许多其他高斯后，某个高斯的重要性会发生变化。留一法重要性是在原始模型中测量的，因此无法反映这种变化。我们直接检验了这一前提，所用的高斯具有相同的留一法重要性。

设计。 我们使用留一法重要性和评分损失，在单个阶段对原始模型进行了剪枝，并考察了被移除的高斯。我们将结果称为剪枝模型。*LMnOP* 保留了其中一部分高斯，并移除了其余高斯。我们将留一法重要性相同的高斯配成对，每对包含一个被*LMnOP*保留的高斯和一个被其移除的高斯。因此，一对中的两个高斯在剪枝前同样不重要，唯一的区别在于*LMnOP*是否保留了它们。如果剪枝没有改变重要性，那么它们在剪枝后仍应同样重要。我们通过使用相同的损失和参考图像进行渲染，在两个模型中测量了这两个高斯的重要性。在原始模型中，我们单独移除该高斯，并记录损失增加。

表 2. **原始模型和剪枝模型中成对高斯的重要性**。每对高斯的留一法重要性相同。*LMnOP* 保留每对中的一个高斯，并移除另一个。数值为中位数（$\times 10^{-6}$）。条形图以共同的刻度显示每一对高斯。

		原始模型		剪枝模型		
移除	场景	Kept	已移除	Kept	已移除	比例
90%	Bicycle	0.035	0.033	**0.167**	0.055	**3.0**×
99%	Bicycle	0.104	0.102	**1.295**	0.191	**6.8**×
	厨房	0.209	0.203	**4.285**	0.805	**5.3**×
	Garden	0.105	0.100	**1.470**	0.291	**5.0**×

在剪枝模型中，我们单独恢复该高斯，并记录损失减少。我们针对每种条件测量了500对样本，这些样本在任何测量之前就已选定。

结果。 在原始模型中，一对高斯的重要性几乎相同，符合预期（表 2）。而在剪枝模型中，它们的重要性则不相同。经过 90% 的移除后，*LMnOP* 保留的高斯比移除的高斯重要 3.0 倍；经过 99% 的移除后，则重要 5.0 到 6.8 倍。在 88–94% 的配对中，保留的高斯更重要。将所有保留的高斯恢复到剪枝模型中，可使峰值信噪比提高 1.7–3.3 dB；而恢复同样数量的已移除高斯，只能使其提高 0.4–0.8 dB。有人可能会问，*LMnOP* 是否只是保留了尺寸更大或不透明度更高的高斯。在配对还按照不透明度、尺寸和可见视图数量进行匹配后，这种差异依然存在。

因此，高斯的重要性取决于哪些高斯仍然保留。留一法重要性在剪枝前测量，无法反映这种变化，而*LMnOP*会保留那些后来变得重要的高斯。该测试考察的是一种在剪枝前仅计算一次的留一法排序。

5.4. 对原始模型的鲁棒性

到目前为止，原始模型均使用 3DGS 进行训练。当多个高斯覆盖同一区域时，LMO重要性与留一法重要性存在差异。高斯如何覆盖场景取决于原始模型的训练方式。因此，我们测试了 *LMnOP* 的优势是否适用于采用不同方式训练的原始模型。我们使用六种方法在 Mip-NeRF360 场景上训练原始模型，并将 *LMnOP* 与五种基线方法（表 3）进行了比较。

训练方法。 3DGS [2] 是参考方法，通过克隆和分裂来增加高斯。其他五种方法分别改变三个属性中的一个。第一个属性是高斯的放置位置。MCMC [22] 通过采样和重新定位来放置高斯。我们将其在每个场景中的高斯数量设置为与 3DGS 相同。因此，这两个模型的差异在于高斯的位置，而不在于高斯的数量。第二个属性是高斯的增长方式。AbsGS [23] 和

表 3。**使用不同方法训练的模型的剪枝**。 在 Mip-NeRF360 上移除 90% 和 99% 高斯后的学习感知图像块相似度（↓），未进行微调。第一行为原始模型。

		训练方法					
方法	Rem.	3DGS	MCMC	FastGS	Mini-Splatting-D	AbsGS	Pixel-GS
原始	0%	0.257	0.224	0.306	0.210	0.229	0.227
EAGLES*	90%	**0.394**	**0.428**	**0.568**	**0.428**	**0.420**	**0.369**
	99%	**0.621**	**0.636**	**0.721**	**0.645**	**0.647**	**0.606**
MesonGS*	90%	**0.408**	0.433	0.583	0.453	0.441	0.382
	99%	0.634	0.638	0.730	0.665	0.658	0.622
GaussianPOP†	90%	0.409	**0.410**	**0.561**	**0.420**	0.455	**0.382**
	99%	**0.628**	**0.620**	**0.712**	**0.643**	**0.653**	**0.612**
RAP	90%	0.468	0.469	0.583	0.472	0.516	0.428
	99%	0.660	0.658	0.732	0.663	0.691	0.638
核心集*	90%	0.412	0.440	0.578	0.468	**0.439**	0.391
	99%	0.634	0.650	0.729	0.667	0.661	0.620
LMnOP（我们的方法）	90%	**0.373**	**0.390**	**0.535**	**0.398**	**0.403**	**0.351**
	99%	**0.594**	**0.590**	**0.714**	**0.622**	**0.626**	**0.575**

†我们的实现。 *仅使用重要性或选择组件。

Pixel-GS [24]改变了添加新高斯的位置规则。Mini-Splatting-D [6] 是 Mini-Splatting 简化之前的稠密模型。这三种方法旨在呈现更精细的细节并实现更好的空间覆盖。第三个属性是模型当前的紧凑程度。FastGS [25]在训练过程中对高斯进行剪枝，因此其模型中的高斯数量远少于其他模型。

结果。 *LMnOP* 在 12 种条件中的 11 种下取得了最低的学习感知图像块相似度。在移除率为 99% 的 MCMC 条件下，它将学习感知图像块相似度从作为最佳基线的 0.620 降至 0.590。在 AbsGS、Pixel-GS 和 Mini-Splatting-D 上的两种剪枝率下，它同样都是最佳方法。因此，*LMnOP* 的优势并不取决于高斯的放置位置或其生长方式。

唯一的例外是 FastGS 在移除率为 99% 时，GaussianPOP 和*LMnOP* 的结果几乎持平，分别为 0.712 和 0.714。这种情况比其他情况极端得多。由于 FastGS 本身已较为紧凑，移除 99% 后每个场景仅剩约 4,000 个高斯，而 3DGS 则约有 34,000 个。高斯数量如此之少时，所有方法的 LPIPS 都高于 0.71，彼此差距至多为 0.020。我们认为，FastGS 的紧凑性导致了优势未能显现。当多个高斯覆盖同一区域时，LMO重要性与留一法重要性有所不同。在训练期间已经经过剪枝的模型中，这类高斯所剩无几。在同一模型上移除 90% 时，*LMnOP* 再次表现最佳，得分为 0.535，而对比方法为 0.561。

5.5. 剪枝率

上面的比较采用了 90% 和 99% 的移除率。我们还在 Mip-NeRF360 上将剪枝率从 50% 调整到 99%（图5）。每个点都表示对原始模型进行的一次独立剪枝。在较低的剪枝率下，大多数方法表现接近，而在移除率达到 80% 之前，GaussianPOP 的峰值信噪比略高。随着移除的高斯数量增加，*LMnOP* 的优势逐渐扩大。从

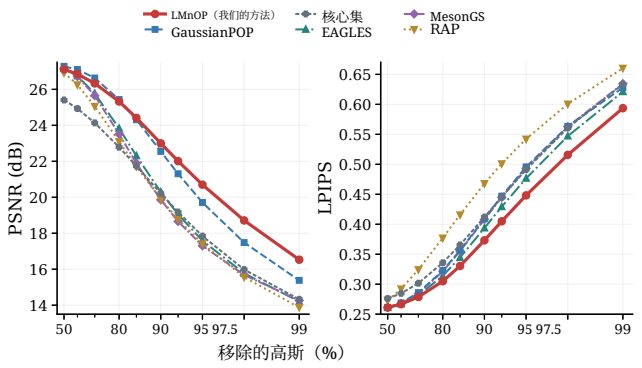


图 5. **不同剪枝比例下的峰值信噪比和学习感知图像块相似度。** Mip-NeRF360，未经微调。每个点表示对原始模型进行的一次单独剪枝。横轴按保留比例的对数刻度间隔排列。

表 4. **剪枝后的微调。** Mip-NeRF360，移除率为 90%。所有方法均使用相同的 5,000 步，并固定高斯数量。每个区块分别排名。

	微调前			5,000 步后		
方法	峰值信噪比↑	结构相似性指数↑	学习感知图像块相似度↓	峰值信噪比↑	结构相似性指数↑	学习感知图像块相似度↓
EAGLES*	**20.29**	**0.651**	**0.394**	**26.22**	**0.778**	**0.321**
MesonGS*	19.86	0.630	**0.408**	25.98	0.769	0.331
GaussianPOP†	**22.55**	**0.664**	0.409	**26.97**	**0.786**	**0.327**
RAP	19.89	0.568	0.468	25.92	0.743	0.374
核心集*	20.22	0.646	0.412	26.08	0.773	0.332
LMnOP（我们的方法）	**23.00**	**0.715**	**0.373**	**27.01**	**0.801**	**0.304**

†我们的实现　　*仅包含重要性或选择组件。

70% 的移除率起，其学习感知图像块相似度最低；从 85% 起，其峰值信噪比最高。其相对于 GaussianPOP 的峰值信噪比优势从移除率为 90% 时的 0.45 dB 扩大到 99% 时的 1.15 dB。这一差距既包含评分损失的影响，也包含重要性的影响，我们将在第5.7节对二者进行分析。

5.6. 剪枝后的微调

上述所有结果均是在未进行微调的情况下测得的。然而，若干剪枝方法是在微调后进行评估的 [1, 7, 8]。因此，我们还对 Mip-NeRF360 上移除率为 90% 的每个模型进行了微调，使用训练图像执行相同的 5,000 个优化步骤。我们禁用了致密化和剪枝，因此高斯的数量保持不变。微调后，*LMnOP*在全部三个指标上仍然表现最佳（表 4）。微调修复了剪枝造成的大部分损失，并缩小了不同方法之间的差异。然而，更优的选择机制使 *LMnOP* 在结构相似性指数和学习感知图像块相似度上的优势仍然清晰可见。

5.7. 消融实验

最后，我们分析*LMnOP* 在 Mip-NeRF360 上不进行微调时的组成部分（表 5）。我们首先考察单个剪枝阶段中基于平方误差的逐一剔除重要性（第一行）。将平方误差替换为我们的评分损失后，在两种剪枝比率下的三个指标均有所提升（第二行）。

表 5。**消融实验。** 未进行微调的 Mip-NeRF360。第一行是基线，即在单个剪枝阶段中基于平方误差计算的留一法重要性；最后一行是 *LMnOP*。

组件			移除 90%			移除 99%		
评分 loss	剪枝 8 次	LMO	峰值信噪比↑	结构相似性指数↑	学习感知图像块相似度↓	峰值信噪比↑	结构相似性指数↑	学习感知图像块相似度↓
留一法，平方误差			20.17	0.639	0.421	13.39	0.370	0.636
✓	–	–	**20.77**	**0.675**	**0.387**	**14.05**	**0.405**	**0.621**
✓	✓	–	**23.08**	**0.699**	**0.389**	**16.81**	**0.470**	**0.594**
✓	–	✓	**23.00**	**0.715**	**0.373**	**16.53**	**0.467**	**0.594**

接着，我们比较了两种处理同时移除多个高斯的方法。使用逐一剔除重要性时，我们分八个阶段进行剪枝，并在每个阶段后再次对剩余高斯进行评分（第三行）。*LMnOP*则使用 LMO重要性，并在完成 R 重要性计算后一次性进行剪枝（第四行）。

评分损失提升了剪枝质量，同时考虑一并移除的高斯的两种方法也提升了剪枝质量。在评分损失相同且仅采用一个剪枝阶段时，LMO重要性比留一法重要性的 PSNR 高出 2.23 和 2.48 dB（第二行和第四行）。*LMnOP* 的表现也接近采用八个阶段的变体。在移除 90% 的高斯时，*LMnOP* 的 SSIM 和 LPIPS 表现更好，PSNR 仅低 0.08 dB。在移除 99% 的高斯时，两者的 LPIPS 相同，而*LMnOP* 的 PSNR 低 0.28 dB，SSIM 低 0.003。

6. 局限性

在 Mip-NeRF360 上，*LMnOP* 在移除 90% 和 99% 的高斯时，每个场景分别耗时 19.5 分钟和 17.0 分钟，而 GaussianPOP 分别耗时 0.9 分钟和 0.6 分钟。造成这一差距的部分原因是评分损失。表5 中的八阶段留一法变体耗时分别为 10.7 分钟和 7.4 分钟。*LMnOP* 的速度约为该变体的一半，因为它在进行 R重要性计算时会保留所有高斯。八阶段变体在移除 99% 的高斯时，PSNR 和 SSIM 也略优于*LMnOP* 。在剪枝比例较低、以及原始模型本身已经较为紧凑时，*LMnOP* 的优势也很小。将 LMO 重要性与多个剪枝阶段结合，将留待未来研究。

7. 结论

我们介绍了 *LMnOP*，它还会在移除其他高斯的情况下估计每个高斯的重要性。其 μ-σ 渲染通过给出每个像素颜色的精确均值和方差，使这一计算变得可行。在不进行微调的情况下，*LMnOP* 在三个数据集、90%和99%移除率下的18项指标中，有16项取得了最佳结果。我们的分析进一步表明，一个高斯的重要性取决于哪些高斯仍然保留。这些结果支持在剪枝预计会产生的子集背景下评估每个高斯。

大语言模型使用声明

我们在本研究中使用了 ChatGPT（OpenAI）和 Claude（Anthropic）。对于文本、相关工作、图表和研究思路，这些工具对作者生成的内容进行了编辑和润色。对于代码和实验，这些工具生成了实现代码，并协助设计、运行和分析实验；作者则审阅、编辑并批准了全部相关工作。我们还将手稿交给这两个工具，以获取审稿人风格的反馈。没有使用大语言模型来创建或标注数据。作者核验了每一项推导、结果和引文，并对本文承担全部责任。

参考文献

[1] Soonbin Lee、Yeong-Gyu Kim、Simon Sasse、Tomas M. Borges、Yago Sanchez、Eun-Seok Ryu、Thomas Schierl 和 Cornelius Hellge。GaussianPOP：通过误差量化实现紧凑3D高斯泼溅的原理化简化框架。arXiv预印本 arXiv:2602.06830，2026年。1、2、5、8[2] Bernhard Kerbl、Georgios Kopanas、Thomas Leimk¨uhler 和 George Drettakis。用于实时辐射场渲染的3D高斯泼溅。《ACM图形学汇刊》，2023年。1、2、3、4、7[3] Joo Chan Lee、Daniel Rho、Xiangyu Sun、Jong Hwan Ko 和 Eunbyung Park。用于辐射场的紧凑3D高斯表示。发表于 CVPR，2024年。1、2[4] Sharath Girish、Kamal Gupta 和 Abhinav Shrivastava。EA-GLES：采用轻量级编码的高效加速3D高斯。发表于 ECCV，2024年。2、5[5] Shuzhao Xie、Weixiang Zhang、Chen Tang、Yunpeng Bai、Rongwei Lu、Shijia Ge 和 Zhi Wang。MesonGS：通过高效属性变换实现3D高斯的后训练压缩。发表于 ECCV，2024年。1、2、5[6] Guangchi Fang 和 Bing Wang。Mini-Splatting：用数量受限的高斯表示场景。发表于 ECCV，2024年。1、2、7[7] Zhiwen Fan、Kevin Wang、Kairun Wen、Zehao Zhu、Dejia Xu 和 Zhangyang Wang。LightGaussian：实现15倍压缩并提高 200+ 帧率的无界3D高斯压缩。发表于 NeurIPS，2024年。2、5、8[8] Alex Hanson、Allen Tu、Vasu Singla、Mayuka Jayawardhana、Matthias Zwicker 和 Tom Goldstein。PUP 3D-GS：面向3D高斯泼溅的原理化不确定性剪枝。发表于 CVPR，2025年。2、5、8[9] Muhammad Salman Ali、Maryam Qamar、Sung-Ho Bae 和 Enzo Tartaglione。去除冗余：通过剪枝高效压缩3D高斯泼溅。发表于英国机器视觉会议，2024年。2、5[10] Kaifa Yang、Qi Yang、Yiling Xu 和 Zhu Li。RAP：用于高效3D高斯泼溅处理的快速前馈、免渲染、属性引导基元重要性分数预测。发表于 CVPR，2026年。2、5[11] Yifei Liu、Zhihang Zhong、Yifan Zhan、Sheng Xu 和 Xiao Sun。MaskGaussian：从概率掩码中获得的自适应3D高斯表示。发表于 CVPR，2025年。2[12] Babak Hassibi 和 David G. Stork。用于网络剪枝的二阶导数：最优脑外科。发表于 NeurIPS，1992年。3[13] Xin Yu、Thiago Serra、Srikumar Ramalingam 和 Shandian Zhe。组合脑外科医生：剪除神经网络中相互抵消的权重。发表于 ICML，2022年。3[14] Zhang Chen、Shuai Wan、Mengting Yu、Fuzheng Yang 和 Junhui Hou。REFINE：通过无渲染基元重要性实现超高效3D高斯泼溅剪枝。arXiv预印本 arXiv:2606.09074，2026年。3[15] Waseem Mousa 和 Alaa Maalouf。通过核心集实现高效3D高斯泼溅的可证明剪枝。发表于 NeurIPS，2026年。3、5[16]Hongbin Ye、Haixu Liu、Shenwei Kang、Wenxiu Wei、Tinglan Wang、Hongdong Zhu、Fan Tang、Qi Gao、Yin Ma、Kai Wen、Hai Wei、Nuo Qian、Wei-Shi Zheng、Shaobo Chen、Fa-Ting Hong 和 Jingke Meng。SplatQ：通过二次选择实现3D高斯泼溅的交互感知剪枝。arXiv预印本 arXiv:2610.13503，2026年。3[17] Jonathan T. Barron、Ben Mildenhall、Dor Verbin、Pratul P. Srinivasan 和 Peter Hedman。Mip-NeRF 360：无界抗锯齿神经辐射场。发表于 CVPR，2022年。4[18] Peter Hedman、Julien Philip、True Price、J an-Michael Frahm、George Drettakis 和 Gabriel Brostow。用于自由视点图像渲染的深度混合。《ACM图形学汇刊》，2018年。4[19] Arno Knapitsch、Jaesik Park、Qian-Yi Zhou 和 Vladlen Koltun。Tanks and Temples：大规模场景重建基准测试。《ACM图形学汇刊》，2017年。4[20] Zhou Wang、Alan C. Bovik、Hamid R. Sheikh 和 Eero P. Simoncelli。图像质量评估：从误差可见性到结构相似性。《IEEE图像处理汇刊》，2004年。4[21] Richard Zhang、Phillip Isola、Alexei A. Efros、Eli Shechtman 和 Oliver Wang。深度特征作为感知度量的惊人有效性。发表于 CVPR，2018年。4[22] Shakiba Kheradmand、Daniel Rebain、Gopal Sharma、Wei-wei Sun、Yang-Che Tseng、Hossam Isack、Abhishek Kar、Andrea Tagliasacchi 和 Kwang Moo Yi。将3D高斯泼溅视为马尔可夫链蒙特卡罗。发表于 NeurIPS，2024年。7[23] Zongxin Ye、Wenyu Li、Sidun Liu、Peng Qiao 和 Yong Dou。AbsGS：恢复3D高斯泼溅中的精细细节。发表于 ACM多媒体，2024年。7[24] Zheng Zhang、Wenbo Hu、Yixing Lao、Tong He 和 Heng-shuang Zhao。Pixel-GS：基于像素感知梯度的3D高斯泼溅密度控制。发表于 ECCV，2024年。7[25] Shiwei Ren、Tianci Wen、Yongchun Fang 和 Biao Lu。FastGS：100秒训练3D高斯泼溅。发表于 CVPR，2026年。7