*Jingfeng*



# NeRF 与 3D 高斯泼溅
# 表示、渲染、压缩与部署

从相机与体渲染，到具有可核查成本的场景表示

Jingfeng · 2026 年 10 月 22 日修订

从正确的重建，走到可信的部署。

一本面向具有基础数学与编程知识读者的学习书：先确定相机与颜色的约定，再推导神经辐射场和高斯投影，随后分析量化、剪枝、熵编码与系统成本。公式、代码与实验分别说明假设和证据；相同的图像误差，并不意味着相同的几何或部署代价。

阅读主线

几何与成像 ⟶ 连续场与显式基元

率失真与编码 ⟶ 评测、诊断与最小实验

基于 2026 年 9 月原稿的完整修订版。

来源核查：2026-10-21 至 22；实验复核：2026-10-22。

# 前言与阅读路线

可信的新视角图像依赖正确的相机、可解释观测的表示、适合当前尺度的渲染，以及可核查的压缩误差与部署成本。本书沿这条链介绍神经辐射场（NeRF）与三维高斯泼溅（3DGS）。读者应熟悉向量、矩阵、导数、概率和基本 Python 编程；遇到协方差、投影 Jacobian 或码率，先检查维度与单位。

本次修订将原稿中后置的 NeRF 基本模型移到方法比较之前，说明体渲染与高斯合成的共同结构及不同含义；压缩从可见误差和实际字节出发，分别衡量基元数、文件大小与每帧成本。推导、论文报告、教学假设和本次实测分别注明。资料截止于 2026-10-22，不表示此前工作已被穷尽。

## 怎样沿着章节前进

第一部分先建立相机与体渲染，再读 NeRF 的网络、位置编码和采样。随后讨论抗混叠、哈希编码与跨场景泛化，再把连续查询换成显式高斯，推导协方差怎样投影成椭圆、方向相关颜色怎样进入合成。

第二部分讨论场景表示的率失真：改变位置、旋转、尺度、不透明度和球谐系数，会怎样影响图像与几何；标量量化、码本、剪枝和熵模型各自节省什么，又引入什么代价。第三部分给出评测协议、研究设计和真正执行过的最小实验。第一次阅读可以略过方法细节，先完成单射线与投影算例；第二次再把论文机制放回这些接口中。

**定义与约定**

全书默认静态场景、针孔相机和已知内外参，除非具体方法另有说明。单场景拟合、跨场景训练和少视图推断是不同实验设定。“测试视角”只用于最终评价；剪枝阈值、码本或位宽的选择使用独立的训练或校准视角。

## 代码与本次实验的边界

代码随书保存在 `experiments/`，配置、运行环境、种子与原始结果保存到本主题的 `output/experiments/`。本次没有发现可用的真实训练 checkpoint，因此压缩 smoke test 使用固定的合成高斯场景与 CPU 教学渲染器。它检查序列化、解码、投影、合成及量化/剪枝误差；真实数据集、训练收敛、GPU 性能和几何重建仍未验证，后续流程另给可操作方案。

本书的压缩对象是场景表示。大模型权重、激活与 KV cache 的低比特压缩，参见丛书《智能体模型量化压缩与部署》；动作条件动力学和闭环规划由《世界模型》展开，机器人控制由《具身智能》展开。本书对相关概念保留必要解释，不依赖其他分册的页码。

## 读完以后应能回答的问题

1. 一条射线使用的参数是相机深度、世界距离还是任意尺度？密度的单位怎样随之改变？
2. 同样采用前到后合成，为什么体密度积分与屏幕高斯仍不能直接视为等价？
3. 一项压缩方案的码本、索引、预测器和元数据有多少真实字节？
4. 画质改善是否来自额外恢复训练，速度改善是否来自不同 renderer？
5. 哪个独立检查能够推翻自己的解释，未见视角又会暴露怎样的失败？

# 目录

**前言与阅读路线** i

## 第一部分　几何、表示与可微渲染 1

**第 1 章　相机几何与体渲染：先把一条射线说清楚** 2

1.1 观测、未知量与可以验证的目标 . . . 2
1.2 固定外参方向与像素约定 . . . 2
1.3 由吸收定律得到像素颜色 . . . 3
1.4 颜色空间是成像模型的一部分 . . . 5
1.5 从训练误差到可信诊断 . . . 5

**第 2 章　NeRF：从连续场到可训练的像素** 6

2.1 网络的分工：位置决定密度，方向参与外观 . . . 6
2.2 先分层覆盖，再用权重引导细采样 . . . 6
2.3 从像素误差传回场参数 . . . 7
2.4 有限区间内可核对的实现骨架 . . . 8
2.5 连续表示的能力边界与下一步 . . . 9

**第 3 章　NeRF 方法地图：尺度、速度、泛化与几何** 10

3.1 抗混叠：像素观察的是区域 . . . 10
3.2 更便宜的场查询：网格、哈希与低秩因子 . . . 11
3.3 减少无效查询：占用结构与提议分布 . . . 11
3.4 跨场景泛化与逐场景稳健拟合 . . . 12
3.5 几何一致性与动态场景各自增加什么约束 . . . 13
3.6 把方法选择写成可证伪的实验 . . . 13

**第 4 章　3D 高斯泼溅：显式基元与可微光栅化** 15

4.1 把合法的几何写进参数化 . . . 15
4.2 球谐颜色包含零阶项 . . . 16
4.3 从三维椭球到二维协方差 . . . 16
4.4 屏幕响应、支持域与排序近似 . . . 17
4.5 前到后合成及其梯度 . . . 18
4.6 初始化、优化与容量控制 . . . 19
4.7 排序、像素覆盖与实际内存 . . . 20

**第 5 章　3DGS 的演化：尺度、结构、泛化与几何** 22

5.1 抗混叠：限制可表示频率，再处理像素足迹 . . . 22
5.2 结构化表示：共享什么，就约束什么 . . . 22
5.3 跨场景泛化：从逐场景优化转向图像条件预测 . . . 23
5.4 几何一致性：2DGS 把基元贴到局部表面 . . . 23
5.5 容量分配的概率视角：MCMC 重定位与位置噪声 . . . 24
5.6 动态场景：时间加入以后，还要保持什么 . . . 25
5.7 把方法选择写成条件与验证 . . . 25

## 第二部分　压缩、率失真与解码系统 27

**第 6 章　从场景表示到率失真与部署预算 28**

6.1 四种成本来自不同阶段 . . . 28
6.2 从 59 个标量建立可核对的基线 . . . 28
6.3 失真到底相对于谁 . . . 29
6.4 率失真目标与属性误差的传播 . . . 29
6.5 一个有头部与码本的算例 . . . 30
6.6 可比较的压缩实验 . . . 30

**第 7 章　高斯属性量化：离散坐标、码本与约束 32**

7.1 标量量化：把零点、偏移与裁剪说清楚 . . . 32
7.2 分组粒度与范围选择 . . . 32
7.3 位置：场景尺度、局部坐标与轮廓 . . . 32
7.4 尺度与旋转：合法性不等于渲染等价 . . . 33
7.5 不透明度与球谐：选择域而非背诵位宽 . . . 34
7.6 向量量化、残差码与真实索引成本 . . . 34
7.7 PTQ、QAT 与混合精度的实验顺序 . . . 35
7.8 混合精度如何与实现相接 . . . 36

**第 8 章　剪枝与压实：从贡献分数到删除风险 37**

8.1 密度控制与部署剪枝解决不同阶段的问题 . . . 37
8.2 删除一个高斯究竟改变什么 . . . 37
8.3 便宜分数的作用与盲区 . . . 38
8.4 重叠、冗余与理论保证的边界 . . . 38
8.5 从全局排序到带保护的渐进剪枝 . . . 39
8.6 删除之后还要重查排序与系统收益 . . . 40

**第 9 章　学习式压缩、熵模型与渐进传输 41**

9.1 量化与熵编码分别改变什么 . . . 41
9.2 从连续分布得到符号概率 . . . 41
9.3 上下文必须先于当前符号到达 . . . 42
9.4 交叉熵、有限精度 CDF 与文件长度 . . . 42
9.5 Anchor、hash context 与混合熵模型 . . . 44
9.6 前馈 codec、图像条件解码与跨场景生成 . . . 45
9.7 渐进码流：可复用的前缀与可渲染的层 . . . 45
9.8 把概率模型写成版本化协议 . . . 46

## 第三部分　实验协议与研究实践 48

**第 10 章　可复现训练、部署评测与排错 49**

10.1 把图像、相机与尺度一起固定 . . . 49
10.2 从训练前的检查到冻结 checkpoint . . . 49
10.3 图像指标的计算对象与平均方式 . . . 49
10.4 几何、时间一致性与任务指标 . . . 50
10.5 存储、内存与时延的测量边界 . . . 50
10.6 用匹配预算的消融回答因果问题 . . . 51
10.7 排错从可证伪的假设开始 . . . 51

**第 11 章 从已核验工作提出研究问题** **53**
11.1 把论文贡献放回统一接口 . . . 53
11.2 可见性分布决定重要性 . . . 53
11.3 理论保证必须带着假设一起阅读 . . . 54
11.4 2025–2026 工作的可核验观察窗口 . . . 54
11.5 三个有明确失败标准的项目 . . . 55
11.6 动态场景、任务与其他分册的连接 . . . 55

**第 12 章 从最小验证到研究原型** **56**
12.1 单射线：让背景项接受手算检查 . . . 56
12.2 投影：解析 Jacobian 与独立数值差分 . . . 56
12.3 固定高斯场景的压缩与剪枝 smoke test . . . 57
12.4 运行与复查记录 . . . 58
12.5 从合成场景走向真实重建 . . . 59
12.6 循序开展的学习计划 . . . 59

**附录 A 符号、单位与实现约定** **61**
A.1 常用公式与量纲检查 . . . 62
A.2 成本量级不等于实测性能 . . . 62

**附录 B 术语与代表练习提示** **63**
B.1 检查答案时应抓住的依据 . . . 63

**参考文献与证据说明** **64**

# 第一部分

# 几何、表示与可微渲染

# 第 1 章　相机几何与体渲染：先把一条射线说清楚

新视角合成从几张照片出发，却不能只在二维图像之间插值。相机移动后，原先被遮挡的部分会出现，近处物体与远处背景的相对位置会改变，同一表面的反光也可能改变。场景表示的任务，是把这些变化解释成一套能从新相机再次成像的三维结构。本章建立全书共同使用的坐标、射线、透射率和颜色约定；下一章据此构造 NeRF，再讨论显式高斯及其压缩。

## 1.1　观测、未知量与可以验证的目标

设训练观测为

$$\mathcal{D} = \{(I_i, K_i, R_{cw,i}, \mathbf{t}_{cw,i})\}_{i=1}^{N},$$

其中图像已经按选定相机模型校正，$K_i$ 为内参，$R_{cw,i}, \mathbf{t}_{cw,i}$ 把世界点变换到第 $i$ 台相机坐标。**新视角合成**要求给定新的相机，预测其图像；**逆渲染**还可能要求分离几何、材质、光照等因素。仅用图像监督的辐射场通常不能唯一恢复这些因素：一个错误表面配合视角相关颜色，也可能重现有限的训练照片。

本书先讨论静态场景、已知相机和固定成像条件。动态、相机误差与曝光变化将在方法章节分别处理。基本任务的测试视角必须独立于训练视角；若目标包含网格重建、测量或机器人感知，还要单独验证几何。训练图像上的低误差只说明模型能解释已见观测。

**定义与约定**

**发射—吸收辐射场**用位置 $\mathbf{x} \in \mathbb{R}^3$ 与单位方向 $\mathbf{d} \in \mathbb{S}^2$ 查询非负密度 $\sigma(\mathbf{x})$ 和颜色 $\mathbf{c}(\mathbf{x}, \mathbf{d})$。这里的密度单位是世界长度的倒数；它控制一段路径中的消光，不是表面占据概率，也不是已经乘过步长的不透明度。该简化模型不显式求解多次散射、折射和光照传输。

NeRF 将这一场参数化为神经网络。标准 3D 高斯泼溅（3DGS）则把显式基元投影到屏幕后合成；它与离散体渲染共享前到后的 alpha 合成结构，但其每个屏幕高斯的不透明度不能直接当作上述物理密度。这个区别决定后文如何解释尺度、剪枝与量化。

## 1.2　固定外参方向与像素约定

本书采用 OpenCV 风格的相机坐标：$+x_c$ 向图像右方，$+y_c$ 向下，$+z_c$ 朝相机前方；三轴构成右手坐标系。世界坐标可以另选，但必须保持一致。外参明确写为

$$\mathbf{x}_c = R_{cw}\mathbf{x}_w + \mathbf{t}_{cw}, \qquad R_{cw}^{\top}R_{cw} = I_3, \quad \det R_{cw} = 1. \tag{1.1}$$

下标 $cw$ 的含义是“输出在相机，输入在世界”，避免把相机到世界矩阵误读成其逆。相机中心 $\mathbf{o}_w$ 满足相机坐标为零，所以

$$\mathbf{o}_w = -R_{cw}^{\top}\mathbf{t}_{cw}, \qquad \mathbf{x}_w = R_{cw}^{\top}\mathbf{x}_c + \mathbf{o}_w.$$

COLMAP 的外参也按世界到相机解释；导入 OpenGL 风格数据时应显式执行轴变换，而不是凭渲染结果反复试负号。OpenCV 的标定与投影文档给出了同一类成像关系 [1]。

对无斜切的针孔相机，内参与投影为

$$K=\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \qquad u=f_x\frac{x_c}{z_c}+c_x, \quad v=f_y\frac{y_c}{z_c}+c_y, \quad z_c>0. \tag{1.2}$$

$f_x, f_y, c_x, c_y$ 的单位均为像素。本书约定整数 $(u,v)$ 表示像素中心，首个像素中心为 $(0,0)$；如果图形接口把中心设为 $(u+1/2, v+1/2)$，需在反投影入口统一转换。两种约定都可使用，混用会产生半像素偏差。

原始照片可能带径向、切向或鱼眼畸变。要么先去畸变并使用相应的新 $K$，要么由完整相机模型反投影，不能把畸变照片直接塞进式 (1.2)。图像缩放和裁剪也必须同步变换内参：若像素齐次坐标满足 $\mathbf{p}'=A\mathbf{p}$，则 $K'=AK$；缩放时的像素中心约定包含在 $A$ 中。

### 1.2.1 从像素得到世界射线

记 $\mathbf{p}=(u,v,1)^\top$，反投影得到

$$\widetilde{\mathbf{d}}_c=K^{-1}\mathbf{p}, \qquad \mathbf{d}_w=\frac{R_{cw}^\top\widetilde{\mathbf{d}}_c}{\|\widetilde{\mathbf{d}}_c\|_2}, \qquad \mathbf{o}_w=-R_{cw}^\top\mathbf{t}_{cw}. \tag{1.3}$$

于是世界射线写成

$$\mathbf{r}(s)=\mathbf{o}_w+s\mathbf{d}_w, \quad s\in[s_n,s_f], \quad \|\mathbf{d}_w\|_2=1. \tag{1.4}$$

这里 $s$ 是沿射线的弧长，单位与场景相同；用 $s$ 而非 $t$ 可以与外参平移区分。查询位置和方向都在世界坐标下。本文将外观函数的方向参数定义为相机指向场景的 $\mathbf{d}_w$；真实光传播至相机的方向相反。若某个 SH 或材质实现要求基元指向相机的方向，必须取负号再查询。

若实现保留未归一化方向 $\widetilde{\mathbf{d}}_w=R_{cw}^\top K^{-1}\mathbf{p}$，并使用 $\mathbf{r}(z)=\mathbf{o}_w+z\widetilde{\mathbf{d}}_w$，则 $z$ 恰为相机前向深度，而非弧长。两个参数满足

$$ds=\|\widetilde{\mathbf{d}}_w\|_2\,dz. \tag{1.5}$$

因此该实现的区间长度必须乘上方向范数。对主光轴以外的像素，直接使用 $\sigma\Delta z$ 会系统性低估光学厚度。NDC 或空间收缩等非线性坐标也不能自动当成物理弧长；应明确变换后的积分规则和场定义。

**直觉**

做一个不依赖训练的检查：令 $R_{cw}=I$、相机中心为 $(0,0,-2)^\top$，则 $\mathbf{t}_{cw}=(0,0,2)^\top$。主点像素的世界射线方向应为 $(0,0,1)^\top$，弧长 $s=2$ 到达世界原点；世界原点投影应恰好是主点。若这三步有一步失败，先修相机约定。

## 1.3 由吸收定律得到像素颜色

把沿射线的密度简写为 $\sigma(s)=\sigma(\mathbf{r}(s))$。在尚未发生终止的条件下，短路径 $ds$ 中的终止概率为 $\sigma(s)ds+o(ds)$。透射率 $T(s)$ 是从近端到 $s$ 仍未被终止的比例，满足

$$T(s+ds)=T(s)(1-\sigma(s)ds)+o(ds).$$

令 $ds\to 0$，得到

$$\frac{dT}{ds}=-\sigma(s)T(s), \qquad T(s_n)=1, \qquad T(s)=\exp\left[-\int_{s_n}^{s}\sigma(\xi)\,d\xi\right]. \tag{1.6}$$

$\sigma \geq 0$ 保证 $T$ 不随距离增长。颜色从每个位置以局部颜色 $\mathbf{c}$ 贡献到相机，得到发射—吸收模型

$$\mathbf{C}(\mathbf{r}) = \int_{s_n}^{s_f} T(s)\sigma(s)\mathbf{c}(\mathbf{r}(s), \mathbf{d}_w)\, ds + T(s_f)\mathbf{c}_{\text{bg}}. \tag{1.7}$$

最后一项是穿过有限积分区间的剩余光。背景可以是常量、已知环境图或另一个可学习远景模型，但必须与训练数据和评测一致。式中没有显式表面法向、反射方程或多次散射，因此拟合的颜色包含拍摄时的综合外观，不能自动解释成反照率。

### 1.3.1 每一段精确积分，整条射线近似求积

将区间分为 $K$ 段 $[b_k, b_{k+1})$，长度 $\delta_k = b_{k+1} - b_k$；在每段内以常数 $\sigma_k, \mathbf{c}_k$ 近似场。进入第 $k$ 段的透射率为 $T_k$。这一段贡献为

$$\int_0^{\delta_k} T_k e^{-\sigma_k \xi} \sigma_k \mathbf{c}_k \, d\xi = T_k(1 - e^{-\sigma_k \delta_k})\mathbf{c}_k.$$

所以

$$\alpha_k = 1 - e^{-\sigma_k \delta_k}, \qquad T_1 = 1, \tag{1.8}$$

$$T_{k+1} = T_k(1 - \alpha_k), \qquad w_k = T_k \alpha_k, \tag{1.9}$$

$$\widehat{\mathbf{C}}(\mathbf{r}) = \sum_{k=1}^{K} w_k \mathbf{c}_k + T_{K+1}\mathbf{c}_{\text{bg}}. \tag{1.10}$$

指数中的光学厚度 $\sigma_k \delta_k$ 无量纲。密度换成不透明度必须同时指定步长；同一 $\sigma$ 的区间缩短一半，$\alpha$ 也会改变。由 $w_k = T_k - T_{k+1}$ 可得

$$\sum_{k=1}^{K} w_k + T_{K+1} = 1. \tag{1.11}$$

它既提供“首个终止位置”的概率解释，也是一条很强的代码检查。离散误差来自段内常量近似；上面的段积分本身是精确的。

### 1.3.2 可以手算的三段教学数据

取 $\delta_1 = \delta_2 = \delta_3 = 1$，令

$$(\sigma_1, \sigma_2, \sigma_3) = (-\ln 0.8, -\ln 0.5, -\ln 0.9),$$

三段颜色依次为红、绿、蓝单位向量，背景为白色 $(1, 1, 1)$。由密度得到 $(\alpha_1, \alpha_2, \alpha_3) = (0.2, 0.5, 0.1)$。

**表 1.1** 三段合成教学算例；这些数值是合成输入，不是场景重建实验。

| 段 $k$ | $\delta_k$ | $\alpha_k$ | 进入透射率 $T_k$ | 权重 $w_k$ |
|---|---|---|---|---|
| 1 | 1 | 0.2 | 1.00 | 0.20 |
| 2 | 1 | 0.5 | 0.80 | 0.40 |
| 3 | 1 | 0.1 | 0.40 | 0.04 |
| 背景 | — | — | 0.36 | 0.36 |

最终线性颜色为

$$\widehat{\mathbf{C}} = (0.20, 0.40, 0.04) + 0.36(1, 1, 1) = (0.56, 0.76, 0.40).$$

若遗漏背景，就会把尚未终止的 36% 光线错误地当作黑色。若误用最后一个进入透射率 $T_3 = 0.40$ 补背景，权重总和会成为 1.04；应使用处理完第三段后的 $T_4 = 0.36$。配套最小实验将用独立的光学厚度表达检查该结果及全透明、首段不透明等边界。

## 1.4 颜色空间是成像模型的一部分

辐射的线性叠加应在线性 RGB 中进行，而 JPEG/PNG 照片常使用 sRGB 等非线性编码。对归一化 sRGB 通道值 $q$，标准解码为 [2]

$$\mathrm{lin}(q) = \begin{cases} q/12.92, & q \le 0.04045, \\ ((q + 0.055)/1.055)^{2.4}, & q > 0.04045. \end{cases} \tag{1.12}$$

将黑与白各混合一半，线性值是 0.5，编码回 sRGB 约为 0.735；直接混合 sRGB 则得到 0.5，明显更暗。这个简单例子说明，色域约定会影响透明边缘和背景合成。

不少公开 NeRF/3DGS 基线直接拟合数据集中归一化后的 sRGB 数值，以保持其既定协议；这种做法可用于图像回归，但不能再宣称合成完全遵循线性辐射传输。复现时应先忠实记录基线约定。若改用线性训练，必须同步处理输入、背景、输出编码、损失和评测，另报可比较的结果，不能只改变其中一项。具有 alpha 通道的输入还要区分直通颜色与预乘 alpha 颜色，避免再次乘 alpha。

在世界长度统一缩放为 $a$ 倍时，相机中心、近远端与几何一并乘 $a$；若希望光学厚度不变，密度应变为 $\sigma/a$。因此单目重建的未知尺度不仅影响几何单位，也影响“密度数值”的解释。3DGS 的尺度属性则随世界尺度一起变化，其独立不透明度通常不按 $1/a$ 换算。

## 1.5 从训练误差到可信诊断

基本的射线损失为

$$\mathcal{L}_{\mathrm{rgb}} = \frac{1}{|\mathcal{R}|} \sum_{\mathbf{r} \in \mathcal{R}} \|\widehat{\mathbf{C}}(\mathbf{r}) - \mathbf{C}^*(\mathbf{r})\|_2^2. \tag{1.13}$$

若每通道数据范围为 $[0, L]$，PSNR 为 $10 \log_{10}(L^2/\mathrm{MSE})$；$L = 1$ 时才可写成 $-10 \log_{10} \mathrm{MSE}$。SSIM 比较局部结构，LPIPS 比较选定网络的特征距离。它们都不能代替表面距离、法向误差或下游任务指标。相同预测图在不同颜色空间、裁剪范围和掩码下也会得到不同数值，评测章将进一步固定这些协议。

开始训练前，先投影一小组点到输入图像，检查相机前方深度符号、主点、图像边界和畸变残差；再查看相机中心及视锥覆盖。开始研究新的渲染器前，先用本章三段数据验证背景和守恒式。这样可以把相机错误、合成错误与模型容量不足逐层分开。

### 练习与提示

1. 对 $K = \mathrm{diag}(2, 2, 1)$、主点为零的相机，像素 $(1, 0)$ 的未归一化方向为 $(0.5, 0, 1)$。计算相机深度区间 $z \in [2, 3]$ 对应的弧长，说明漏掉范数后密度的偏差。提示：使用式 (1.5)。
2. 把三段教学数据的第二段等分为两段，保持密度、颜色不变，证明最终颜色不变。提示：$(1 - \alpha_{\mathrm{half}})^2 = 1 - \alpha_2$；这比只核对固定数字更能检验实现。
3. 推导射线累计不透明度 $A = 1 - T_{K+1}$，并解释 $\sum_k w_k s_k$ 与条件深度 $\sum_k w_k s_k / A$ 的区别。提示：前者未对终止于区间内这一事件归一化；$A = 0$ 时后者无定义。
4. 若训练视图清晰而新视图出现漂浮薄片，列出至少两个几何/相机原因和一个外观原因，再设计各自的区分检查。不要只增加网络容量。

# 第 2 章　NeRF：从连续场到可训练的像素

有了第 1 章的相机射线与合成公式，NeRF 的核心问题变成：用什么函数表示每个位置的密度与颜色，又怎样让有限的网络查询尽量落在有用的位置？本章先完整建立一个单场景 NeRF 的训练过程，再进入抗混叠、网格加速和泛化方法。这里的“连续”指可以在任意坐标查询，不代表训练能够恢复任意细小的结构。

## 2.1　网络的分工：位置决定密度，方向参与外观

原始 NeRF[3] 使用多层感知机（MLP）。一个共享位置分支输出原始密度及特征，再由方向条件的颜色分支预测 RGB：

$$(a_\sigma, \mathbf{h}) = F_{\theta_x}(\gamma_x(\mathbf{x})), \qquad \sigma = \phi(a_\sigma), \qquad \mathbf{c} = F_{\theta_c}(\mathbf{h}, \gamma_d(\mathbf{d}_w)). \tag{2.1}$$

$\phi$ 必须非负，常见选择为 ReLU 或 softplus；颜色输出常用 sigmoid 约束在 $[0,1]$。原始架构以多层、宽度为 256 的位置 MLP 为主体，具有位置编码跳接，并在较小的方向分支中得到颜色。复现时应记录实际层数、宽度、跳接位置和激活；“使用 NeRF”不足以描述网络配置。

让密度仅依赖位置，是把静态几何在不同相机之间共享的归纳偏置。让颜色依赖方向，则可拟合拍摄条件下的镜面亮斑等非朗伯外观。但这并不等于显式求解材质与照明：网络可能用过强的方向分支掩盖错误几何，未观测方向也缺乏约束。为动态场景加入时间之前，应先说明哪些属性随时间变化，不能简单让所有输入自由耦合。

### 2.1.1　位置编码改变可学习的频率

低维坐标直接输入普通 ReLU MLP，往往优先拟合低频。对归一化标量坐标 $p$，本书使用如下 Fourier 编码约定：

$$\gamma_L(p) = \big(p, \sin(\pi p), \cos(\pi p), \dots, \sin(2^{L-1}\pi p), \cos(2^{L-1}\pi p)\big). \tag{2.2}$$

三维坐标逐维编码后维数为 $3(1+2L)$。不同代码可能不附加原坐标、或不写 $\pi$；这些约定在相应坐标缩放后可以对应，但不能在比较时无声互换。位置通常比方向使用更高频率，因为细纹理与边缘需要空间高频，而方向过高频容易记住训练相机。

假设一个坐标在 $[-1,1]$ 范围内，$L=4$ 的最高频项为 $\sin(8\pi p)$，跨整个范围经历八个周期；若把同一物体缩到 $[-0.1, 0.1]$ 而不调整频率，编码实际覆盖的周期便减少。由此可见，编码带宽、场景归一化和采样间距是连在一起的。提高 $L$ 不能弥补相机覆盖不足，也不能绕过像素本身的有限分辨率。

**适用条件与误区**

密度激活不是随手更换的选项。ReLU 的负区间梯度为零，softplus 的密度底值又可能制造空空间雾。初始化、激活偏置、近远端和背景应一起检查；如果前方密度过早饱和，后方结构会几乎收不到梯度。

## 2.2　先分层覆盖，再用权重引导细采样

沿每条射线固定查询大量等距点，空空间也消耗网络运算。原始 NeRF 用粗、细两个网络分配查询：粗网络建立近似可见性分布，细网络将更多点放到该分布提示的重要区域。这里的“分层采

样”先指把区间分层后在各层内随机取点，不应与后续的“层次式粗到细网络”混为一个操作。

取边界 $b_1 < \cdots < b_{K+1}$，在第 $k$ 段采样

$$s_k = b_k + \xi_k(b_{k+1} - b_k), \qquad \xi_k \sim \mathcal{U}[0,1).$$

测试时可用固定中点以消除采样噪声。该分层策略保证每个区间都被覆盖，并在训练中抖动查询位置。粗网络由式 (1.10) 得到非负权重 $w_k$；它表示射线在该区间终止的近似概率质量，而不是密度 $\sigma_k$ 本身。

### 2.2.1 区间概率质量不是概率密度

令 $\Delta_k = b_{k+1} - b_k$，$S = b_{K+1} - b_1$，加入少量均匀探索质量：

$$p_k = \frac{w_k + \varepsilon\Delta_k/S}{\sum_j w_j + \varepsilon}, \qquad \rho(s) = \frac{p_k}{\Delta_k} \quad (s \in [b_k, b_{k+1})). \tag{2.3}$$

$p_k$ 才是区间被选中的概率，$\rho$ 才是对弧长的概率密度。等宽区间中加入同样的 $\varepsilon$ 也常见；不等宽时，式 (2.3) 的好处是全空射线退化为对弧长均匀采样，而不是对不同宽度的区间等概率采样。

定义累积质量 $F_0 = 0$、$F_k = \sum_{j\le k} p_j$。对 $u \sim \mathcal{U}[0,1)$，找到 $F_{k-1} \le u < F_k$，逆累积分布函数（inverse CDF）给出

$$s = b_k + \frac{u - F_{k-1}}{p_k}\Delta_k. \tag{2.4}$$

若实现的概率质量为零，要跳过零宽 CDF 区间并保护除法。教学例：边界为 $(0,1,3)$，区间质量为 $(1/4, 3/4)$，则 PDF 高度为 $(1/4, 3/8)$；$u = 0.1$ 得到 $s = 0.4$，$u = 0.625$ 得到 $s = 2$。直接把区间质量当 PDF 高度会得到错误的区间内分布。

原始 NeRF 代码常用粗样本的相邻中点作为重采样箱边界，并避开端部权重；这与本节显式边界的教学记法不同。重现论文时应保留其边界构造。无论采用哪一种，新增细样本与粗样本合并排序后，都必须重新计算每个样本代表的积分间距，不能沿用粗阶段的 $\delta_k$。

### 2.2.2 采样分布与梯度的边界

原始两网络训练通常停止通过重采样权重和逆 CDF 的梯度：先对粗权重 detach，再决定细样本位置。细网络损失更新细网络，粗网络由自己的颜色损失训练。这样避免把离散区间查找当成普通光滑操作，也防止把“改采样位置”和“改场值”的梯度混在一起。它不意味着所有可学习采样器都必须停止梯度；采用可重参数化提议网络时，需要另外定义估计器和训练目标。

粗权重只描述当前模型认为有可见贡献的位置，并非真实几何标签。若薄结构在粗查询中完全漏掉，重要性采样也可能继续忽略它；保留覆盖全区间的粗样本、探索质量和合理的初始密度，有助于避免这种自我强化。

## 2.3 从像素误差传回场参数

忽略采样位置对参数的依赖，颜色对样本颜色的导数为

$$\frac{\partial \widehat{\mathbf{C}}}{\partial \mathbf{c}_k} = w_k I_3.$$

对密度的导数更能解释遮挡。令 $\mathbf{B}_{k+1}$ 为从第 $k+1$ 段开始、以透射率一重新合成的后方颜色，包含背景，则

$$\widehat{\mathbf{C}} = \mathbf{C}_{<k} + T_k\{\alpha_k \mathbf{c}_k + (1-\alpha_k)\mathbf{B}_{k+1}\}.$$

由 $\partial\alpha_k/\partial\sigma_k = \delta_k(1-\alpha_k)$，可得

$$\frac{\partial \widehat{\mathbf{C}}}{\partial \sigma_k} = T_k\delta_k(1-\alpha_k)(\mathbf{c}_k - \mathbf{B}_{k+1}). \tag{2.5}$$

它揭示三个现象。前方不透明导致 $T_k$ 小，后方难学；本段已饱和导致 $1-\alpha_k$ 小，本段密度也难改；本段颜色与后方相同，则仅凭这条射线几乎不能决定密度应放在哪里。因此“有梯度”与“几何可辨识”并不相同，多视角覆盖和适当先验仍然必要。

训练目标一般为粗、细网络损失之和

$$\mathcal{L} = \|\widehat{\mathbf{C}}_c - \mathbf{C}^*\|_2^2 + \|\widehat{\mathbf{C}}_f - \mathbf{C}^*\|_2^2,$$

并对随机采样的射线求平均。随机像素给出图像损失的随机梯度；它不要求从同一张图像连续遍历全部像素。对稀疏视角、曝光不一致等情况，不能只把学习率调小来代替重新建模。

## 2.4 有限区间内可核对的实现骨架

下面明确使用 $K+1$ 个边界、$K$ 组场值，所有方向已归一化，且保持有限远端。它是对应本书离散模型的概念代码；张量批处理、位置编码和网络查询在外部完成。

**Listing 2.1** 有限区间的前到后合成；背景使用处理完全部样本后的透射率。

```python
def composite(edges, sigma, rgb, background):
    # edges: [K+1], sigma: [K], rgb: [K, 3]
    delta = edges[1:] - edges[:-1]
    assert (delta > 0).all()
    tau = sigma.clamp_min(0) * delta
    alpha = -torch.expm1(-tau)
    survival = torch.exp(-tau)
    trans = torch.cumprod(torch.cat([
        torch.ones_like(survival[:1]), survival
    ]), dim=0)  # [K+1]
    weights = trans[:-1] * alpha
    color = (weights[:, None] * rgb).sum(0)
    color = color + trans[-1] * background
    return color, weights, trans[-1]

def proposal_masses(edges, weights, eps=1e-5):
    width = edges[1:] - edges[:-1]
    exploration = width / width.sum()
    masses = weights.detach() + eps * exploration
    return masses / masses.sum()
```

$-\mathrm{expm1}(-\tau)$ 在 $\tau$ 很小时比直接计算 $1-e^{-\tau}$ 更稳定。累乘的最后一个元素具有背景语义，不能用最后一个样本的进入透射率替代。更深的累乘可以累计光学厚度后取指数，极不透明时的下溢对应近零透射率；训练时仍应避免未经分析的裁剪引入梯度异常。

教学实现不把末段强行设成无限长。原始 NeRF 的一些代码使用一个很大的末段长度，使最后一个正密度样本吸收所有剩余透射率；这是一种边界约定，而非所有场景必需的积分定律。透明场景、显式背景或有限包围盒中应检查它是否符合任务。细阶段也可由已排序的 $s_k$ 构造相邻中点分区并固定两端为 $s_n, s_f$，从而让每个样本拥有明确的有限代表区间。

### 2.4.1 最小调试次序

先固定网络输出为常量，验证同密度区间细分前后颜色不变；再使用第 1 章的三段数据检查权重守恒和背景；随后对单个密度做有限差分，核对式 (2.5)。这些检查不依赖训练质量，能定位合成算子错误。最后才训练一个小场景，查看训练与验证误差、深度图、累计不透明度以及粗细采样直方图。

## 2.5 连续表示的能力边界与下一步

原始 NeRF 通过坐标共享参数，可表达视角相关外观并接受图像监督；代价是每个像素要反复查询场。若一张 $H \times W$ 图有 $K_c$ 个粗样本和 $K_c + K_f$ 个细样本，网络评估量随 $HW(2K_c + K_f)$ 增长，实际时间还取决于批处理、网络宽度、编码和内存读写。

这引出三个不同的问题：空空间查询过多，需要改采样；每次查询昂贵，需要改表示或实现；有限观测有歧义，需要改监督与先验。它们分别通向占用网格/提议采样、显式网格/哈希编码，以及稀疏视角/几何/动态建模。第 3 章将按这些问题组织方法，避免把所有改进混成单一的速度榜。

**练习与提示**

1. 将式 (2.2) 的位置频数设为 $L_x = 10$，方向频数设为 $L_d = 4$，计算两个编码维数。提示：本书编码包含原坐标，若代码不包含则每个结果减三。
2. 两个重采样区间宽度分别为 1 和 4，当前粗权重均为零。证明式 (2.3) 退化为对整个区间均匀采样；比较直接给每箱相同质量的结果。
3. 用本章梯度公式说明：前方红色样本与后方白背景之间，增加密度会分别怎样改变 R、G、B 通道。提示：看 $\mathbf{c}_k - \mathbf{B}_{k+1}$，不要只说“更不透明”。
4. 设计一个宽度小于粗采样间距的密度尖峰，比较固定等距采样、分层抖动和粗到细采样可能漏检的条件。先给出覆盖概率或反例，再考虑实验，不要预设重要性采样总是更好。

**本章回顾**

NeRF 将位置密度、方向颜色与可微合成接在一起。位置编码控制可表达频率；粗到细采样用当前可见贡献重新分配查询；停止梯度明确了采样器与场优化的边界。一个正确的实现必须同时解释单位射线、区间宽度、末段、背景和颜色空间，而不是只写出一个 MLP。

# 第 3 章　NeRF 方法地图：尺度、速度、泛化与几何

NeRF 的后续研究没有共同的单一终点。有些方法解决远近尺度变化，有些减少场查询，有些从跨场景数据中学习先验，还有些优先得到可用的表面。选择方法之前，应先指出误差发生在哪个环节：相机成像、空间表示、采样分配，还是观测本身不足。本章据此整理代表工作，不把不同数据和硬件下的数字排成性能顺序。

## 3.1　抗混叠：像素观察的是区域

第 2 章使用一条理想射线代表一个像素。实际像素覆盖有限面积，沿深度形成一个锥体截段；相机退远或缩小分辨率时，像素覆盖的世界区域会扩大。点查询仍输出高频纹理，便可能产生摩尔纹或随相机抖动的闪烁。训练误差低也不能排除这一问题，因为训练时只见到某些采样尺度。

Mip-NeRF[4] 用高斯近似锥台内的位置分布，将点位置编码换成积分位置编码。若 $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$，对频率向量 $\boldsymbol{\omega}$ 有

$$\mathbb{E}[\sin(\boldsymbol{\omega}^\top \mathbf{x})] = e^{-\boldsymbol{\omega}^\top \Sigma \boldsymbol{\omega}/2} \sin(\boldsymbol{\omega}^\top \boldsymbol{\mu}), \tag{3.1}$$

$$\mathbb{E}[\cos(\boldsymbol{\omega}^\top \mathbf{x})] = e^{-\boldsymbol{\omega}^\top \Sigma \boldsymbol{\omega}/2} \cos(\boldsymbol{\omega}^\top \boldsymbol{\mu}). \tag{3.2}$$

这一结果来自高斯分布的特征函数。它说明区域越大，编码中的高频衰减越强，而不是让网络自己从混叠数据中猜测应当模糊多少。

例如在一维中取 $\omega = 8\pi$、$\mu$ 固定。标准差从 0.01 增加到 0.1 时，衰减因子从约 0.969 降至约 0.0425。这只是解析教学算例，不是画质测量；它清楚展示了频率与足迹尺度的耦合。对编码取期望并不等于对完整非线性网络和遮挡做精确像素积分，锥台的高斯近似也有误差。

### 3.1.1　无界场景与集中权重的正则

Mip-NeRF 360[5] 处理含远景的无界场景。其空间收缩的一种写法为

$$\operatorname{contract}(\mathbf{x}) = \begin{cases} \mathbf{x}, & r \le 1, \\ (2 - 1/r)\mathbf{x}/r, & r > 1, \end{cases} \qquad r = \|\mathbf{x}\|_2. \tag{3.3}$$

收缩把无限远映入有界区域，给有限网络/网格分配表示能力；它不消除远处的歧义。位置高斯经过该变换时，其协方差需要按局部 Jacobian 传播；射线距离的参数化也应沿用方法的具体定义，不能把收缩后坐标差直接当成世界弧长。

该方法的 distortion 正则鼓励同一条射线的贡献集中。设 $u$ 是所选归一化射线距离，区间 $[u_i, u_{i+1}]$ 宽度为 $\Delta_i$，中心为 $\bar{u}_i$，总概率质量为 $w_i$；将区间内权重密度视为常量。两个独立位置的加权距离积分可化为

$$\mathcal{L}_{\text{dist}} = \sum_{i,j} w_i w_j |\bar{u}_i - \bar{u}_j| + \frac{1}{3} \sum_i w_i^2 \Delta_i. \tag{3.4}$$

对两个互不重叠区间，绝对距离的期望等于中心距离；对同一长度为 $\Delta_i$ 的均匀区间，$\mathbb{E}|U - V| = \Delta_i/3$，故第二项不能遗漏。仅保留第一项会使一个很宽的单一区间具有零正则，违背集中贡献的目的。

正则并不监督真实表面位置。透明介质、多层薄结构和真实半透明外观可能本来就有分散权重；强行集中会引入偏差。改变距离参数 $u$ 或权重定义后，也需要重新解释正则强度。

## 3.2 更便宜的场查询：网格、哈希与低秩因子

连续查询接口不要求由大 MLP 独立完成。网格顶点保存密度或特征，再通过插值得到连续值，便能将一部分计算转换成存储与读写。

Instant-NGP[6] 用多分辨率哈希网格编码位置，配合小网络和高度融合的 GPU 实现。令第 $\ell$ 层分辨率为 $N_\ell$，$\mathbf{v} = \lfloor N_\ell \mathbf{x} \rfloor$，单元内坐标为 $\boldsymbol{\eta} = N_\ell \mathbf{x} - \mathbf{v}$。三线性插值明确写为

$$\mathbf{f}_\ell(\mathbf{x}) = \sum_{\mathbf{b}\in\{0,1\}^3} \left[ \prod_{d=1}^{3} \eta_d^{b_d} (1-\eta_d)^{1-b_d} \right] E_\ell[h_\ell(\mathbf{v}+\mathbf{b})]. \tag{3.5}$$

每层读取单元八个角点的表项，拼接各层特征再交给网络。哈希表限制存储，但碰撞意味着多个顶点共享参数；多尺度有助于降低冲突的破坏，并不保证冲突被完全消除。速度还来自 CUDA 融合与采样结构，不能仅靠把正弦编码换成字典就预期相同结果。

DVGO[7] 直接优化密度和特征体素网格，并以浅层颜色网络保留方向效应；Plenoxels[8] 用稀疏体素密度与球谐系数表示外观，不依赖颜色 MLP。稀疏网格省去空区域，却仍要付出拓扑、索引和插值读取的成本。它们保持发射—吸收渲染语义，与后文在屏幕上合成高斯的接口仍有区别。

TensoRF[9] 把三维张量改写成较低秩因子。CP 型与向量—矩阵型是两种不同设计，例如

$$\mathcal{V}_{\mathrm{CP}}(x,y,z) = \sum_{r=1}^{R} v_r^x(x) v_r^y(y) v_r^z(z), \tag{3.6}$$

$$\mathcal{V}_{\mathrm{VM}}(x,y,z) = \sum_{r=1}^{R} \left[ M_r^{xy}(x,y) v_r^z(z) + M_r^{xz}(x,z) v_r^y(y) + M_r^{yz}(y,z) v_r^x(x) \right]. \tag{3.7}$$

这里的连续函数值通过一维/二维因子插值得到。若每轴分辨率为 $n$，忽略通道与边界，稠密标量网格存 $n^3$ 个数，CP 约存 $3nR$，VM 约存 $3n^2R + 3nR$。这说明低秩结构为何能省存储，也说明复杂非可分结构需要更高秩。不能据此直接推出渲染时间按同比例缩短。

Zip-NeRF[10] 将网格表示与抗混叠结合，对像素足迹覆盖的空间进行有结构的采样/聚合。它提醒我们：把 MLP 换成高速网格以后，像素有限面积的问题仍在；速度改进与成像改进可以相互组合。

## 3.3 减少无效查询：占用结构与提议分布

NerfAcc[11] 将高效采样和体渲染整理为可复用组件。占用网格依据近期密度/不透明度估计，跳过看似空的单元；提议分布则预测射线上更可能贡献的区域。它们对不同问题做近似：前者接近“这里是否值得进入”，后者接近“有限查询应该怎样分配”。

实际占用值往往由有限点抽样、滑动更新和阈值决定，并不普遍构成严格的密度上界。若初始化过稀、更新过慢或阈值过高，细杆和薄叶可能被错判为空，而且训练再也查询不到它们。因此需要预热、更新覆盖、保守阈值或探索策略；跳过空空间的效率必须和漏检一起验证。

如果确实知道一段的密度上界 $\sigma_{\max}$，长度为 $\Delta$，那么该段可能造成的终止质量最多为

$$T_{\mathrm{in}}(1 - e^{-\sigma_{\max}\Delta}) \le T_{\mathrm{in}} \sigma_{\max} \Delta.$$

在每通道颜色和后方颜色均属于 $[0,1]$ 时，这也给出跳过该段的逐通道误差上界。只有“上界成立”的前提得到保证，才可以把这种估计作为误差证书；启发式占用缓存不能直接继承这一保证。

**Listing 3.1** 采样器诊断骨架：同时记录查询量与被漏掉的贡献。

```
for ray in fixed_validation_rays:
    dense = render_with_dense_samples(ray, field)
    fast, stats = render_with_accelerator(ray, field)
    record(error=max_abs(fast - dense),
           queries=stats.num_field_queries,
           skipped=stats.skipped_cells)
# Sweep thresholds on the same frozen field and camera set.
# Do not retrain the field between these measurements.
```

这个诊断固定场参数，因此可把采样误差与训练差异分开。稠密参考本身也应通过进一步细分检查收敛，不能默认有限的“dense”就是真积分。若只比较最终训练 PSNR，则占用漏检可能已经改变场的学习过程，难以确定误差来自哪里。

## 3.4 跨场景泛化与逐场景稳健拟合

pixelNeRF[12] 将输入图像的卷积特征投影/查询到三维位置，利用跨场景学习的条件先验从一张或少量照片预测辐射场；IBRNet[13] 利用源视图特征与多视图聚合预测目标射线外观。这里训练集合包含多个场景，测试一个新场景的推理与逐场景微调应分开计时。相机已知与否、源视图选择和分布偏移都会影响泛化。

RegNeRF[14] 的主要设定不同：它在稀疏输入下优化目标场景，并利用未观测视图的渲染深度平滑等几何正则以及外观先验稳定学习。不能因为名字出现在稀疏视角讨论里，就认定它与前馈跨场景方法拥有相同的训练过程。

NeRF in the Wild[15] 面对同一地标的无约束照片，以图像相关的外观变量和瞬态建模处理曝光、光照与遮挡差异；它不是无需目标场景拟合的通用场景重建器。BARF[16] 则将位姿与场联合优化，通过由低到高开放位置编码频率缓解相机对齐困难。它依赖具体初始化、场景和优化条件，不能理解为任意坏位姿都能自动修复。

**表 3.1** 几种经常被混称为“泛化”的设定；该表比较假设，不比较速度。

| 问题 | 学到或优化的对象 | 需要注明的输入条件 |
|---|---|---|
| 跨场景条件推理 | 多场景共享网络，输入图像特征 | 训练分布、源视图数、相机、是否微调 |
| 稀疏逐场景重建 | 目标场景参数与几何正则 | 目标图数、虚拟视图、先验来源 |
| 无约束照片拟合 | 同一地点的场与外观/瞬态变量 | 照片时间、曝光、遮挡、位姿质量 |
| 位姿与场联合优化 | 相机参数和场参数 | 初值、尺度自由度、畸变及优化范围 |

## 3.5 几何一致性与动态场景各自增加什么约束

若任务是获得表面，任意非负密度的阈值等值面未必稳定。NeuS[17] 用带符号距离函数（SDF）$f(\mathbf{x})$ 的零水平集表示表面，并设计与 SDF 对应的体渲染权重以降低表面定位偏差。SDF 的典型正则为

$$\mathcal{L}_{\text{eik}} = \mathbb{E}_{\mathbf{x}}(\|\nabla f(\mathbf{x})\|_2 - 1)^2. \tag{3.8}$$

它要求函数局部像距离，而不是把任意密度的高值解释成表面。SDF 适合清晰边界的表面；半透明介质、植被和复杂非表面现象未必符合这一假设。不同方法的网格提取阈值、裁剪范围和几何配准也要固定后再比较。

动态场景要查询 $F(\mathbf{x}, \mathbf{d}, t)$，其中 $t$ 现在表示时间，不再表示射线距离。D-NeRF[18] 采用形变场和规范空间场的分工，可概念化为

$$\mathbf{x}' = \mathbf{x} + \Delta_{\phi}(\mathbf{x}, t), \qquad (\sigma, \mathbf{c}) = F_{\theta}(\mathbf{x}', \mathbf{d}).$$

形变把时间信息压入共享结构，避免逐帧完全独立建模。但这种用于图像拟合的坐标形变并不自动守恒真实介质质量，也不能保证未见时刻的动力学。一个移动单目相机在每个时刻只看到一张图像时，运动与视角变化尤其容易混淆；同步多视角、时序正则和已知静态区域可提供额外约束。

本书把动态表示解释到“怎样建模与渲染时空观测”为止。动作条件预测、闭环规划和机器人控制分别属于世界模型与具身智能分册；把渲染时间参数加入模型，并不足以得到可用于控制的世界模型。

## 3.6 把方法选择写成可证伪的实验

若现象是远距离纹理闪烁，先固定场景与相机路径，改变输出分辨率和拍摄尺度，比较点采样与足迹处理；若现象是训练慢，分别统计场查询数、单次查询耗时和数据传输；若稀疏输入导致几何崩坏，就固定输入视角，再验证先验是否真的改善未观测区域。每一步都只改变与假设对应的因素。

**表 3.2** NeRF 家族按干预对象组织的阅读地图。

| 干预对象 | 代表工作 | 验证重点 |
| --- | --- | --- |
| 像素/区域积分 | Mip-NeRF、Zip-NeRF | 训练与测试尺度、混叠和模糊的取舍 |
| 无界空间与权重 | Mip-NeRF 360 | 距离参数、远景覆盖、distortion 强度 |
| 查询表示 | Instant-NGP、DVGO、Plenoxels、TensoRF | 内存、插值、碰撞/秩与实测耗时 |
| 查询位置 | NerfAcc | 漏采样、查询数量、占用更新开销 |
| 观测与先验 | pixelNeRF、IBRNet、RegNeRF | 训练分布、目标视图数、逐场景适配 |
| 几何与时间 | NeuS、D-NeRF | 表面假设、几何误差、时间留出测试 |

### 练习与提示

1. 从高斯特征函数推导式 (3.2)，并说明协方差各向异性时不同方向的高频为何衰减不同。提示：将正弦和余弦看作复指数的虚部与实部。
2. 对只有一个宽度 $\Delta$、质量 $w$ 的区间，求 distortion loss；再把区间均匀切为两半并按质量均分，验证总值相同。提示：跨区间项和内部项都需要计入。
3. 设计一条只在狭窄区间非零的密度函数，说明启发式 occupancy 的有限更新为什么不能证明该单元为空；提出一个能显示漏检的冻结场测试。
4. 在相同稀疏输入下比较跨场景条件模型与逐场景优化模型，列出需要分别计入的训练、预训练、微调和推理成本。提示：同时给出新场景首次可用时间和离线预训练成本。

### 本章回顾

抗混叠改变每次查询代表的区域；网格、哈希与低秩因子改变存储和计算分配；采样器减少无效查询；泛化、几何和动态方法改变监督及先验。它们各自拥有适用假设，组合时还会引入新的代价。下一章转向显式高斯，保留同样的几何与成像纪律，并检查其光栅化近似。

# 第 4 章　3D 高斯泼溅：显式基元与可微光栅化

NeRF 在射线上的许多位置查询连续场；3D Gaussian Splatting（3DGS）则维护一组显式基元，把它们投影为屏幕上的椭圆，再按可见性顺序合成 [19]。二者都能利用图像误差反向优化场景，但计算图、存储对象和近似误差不同。本章依次回答四个问题：怎样保证基元的几何合法，三维协方差怎样投影，颜色与梯度怎样合成，以及何时应该改变基元数量。

## 4.1 把合法的几何写进参数化

第 $i$ 个基元的属性记为

$$\mathcal{G}_i = \{\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, \alpha_i, \mathbf{h}_i\}, \qquad i = 1, \dots, N, \tag{4.1}$$

其中 $\boldsymbol{\mu}_i \in \mathbb{R}^3$ 是世界坐标均值，$\boldsymbol{\Sigma}_i \in \mathbb{R}^{3\times 3}$ 是对称正定协方差，$\alpha_i \in [0,1]$ 是基元不透明度，$\mathbf{h}_i$ 是方向颜色系数。空间响应写为

$$G_i(\mathbf{x}) = \exp\!\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^{\top}\boldsymbol{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)\right]. \tag{4.2}$$

这里没有概率密度的归一化因子 $1/\sqrt{(2\pi)^3 \det \boldsymbol{\Sigma}_i}$。它是一个峰值为 1 的基函数；增大尺度会扩大支持范围，并不会自动降低峰值。标准 3DGS 的 $\alpha_i$ 也不是 NeRF 的消光密度 $\sigma$，不能脱离投影与合成规则，把它换算成某个唯一的物理体密度。

直接更新协方差的六个独立元素，可能使其出现非正特征值。原方法改用旋转和正尺度：

$$\boldsymbol{\Sigma}_i = R(\mathbf{q}_i)\,\mathrm{diag}(s_{i1}^2, s_{i2}^2, s_{i3}^2)R(\mathbf{q}_i)^{\top}, \qquad s_{ij} = \exp(\ell_{ij}) > 0. \tag{4.3}$$

$\ell_{ij}$ 是优化器实际保存的 log 尺度；$s_{ij}$ 是沿局部主轴的标准差，而非方差。它的单位与世界坐标一致，协方差元素的单位则是长度平方。不透明度可由无约束参数 $\tilde{\alpha}_i$ 经 sigmoid 得到，使有限输入对应 $0 < \alpha_i < 1$。

本书约定四元数按实部在前保存：$\mathbf{q} = (w, x, y, z)^{\top}$，由局部基元坐标主动旋转到世界坐标。每次使用前先令

$$\mathbf{q} = \frac{\tilde{\mathbf{q}}}{\|\tilde{\mathbf{q}}\|_2}, \quad \|\tilde{\mathbf{q}}\|_2 > 0,$$

再计算

$$R(\mathbf{q}) = \begin{bmatrix} 1-2(y^2+z^2) & 2(xy-wz) & 2(xz+wy) \\ 2(xy+wz) & 1-2(x^2+z^2) & 2(yz-wx) \\ 2(xz-wy) & 2(yz+wx) & 1-2(x^2+y^2) \end{bmatrix}. \tag{4.4}$$

此时 $R^{\top}R = I$、$\det R = 1$，且 $\mathbf{q}$ 与 $-\mathbf{q}$ 表示同一旋转。零范数四元数没有合法方向；导入或解码时应检查并按约定拒绝或修复，不能仅在分母加一个数后假定所得矩阵仍正交。

任意非零 $\mathbf{v}$ 满足

$$\mathbf{v}^{\top}\boldsymbol{\Sigma}_i\mathbf{v} = \sum_{j=1}^{3} s_{ij}^2 \big(R^{\top}\mathbf{v}\big)_j^2 > 0,$$

因此有限的 log 尺度与合法旋转从结构上保证正定。不过浮点计算仍可能因 $\exp(\ell)$ 上溢、下溢或极大轴比而失稳。实现应检查有限性、尺度范围和条件数，量化后也要重新归一化旋转。尺度在 log 域编码、旋转按流形约束处理的理由，由此已经出现。

## 4.2 球谐颜色包含零阶项

采用第 1 章的方向约定。相机中心为 $\mathbf{o}_w$，颜色查询方向取相机指向基元的单位世界方向

$$\mathbf{d}_i = \frac{\boldsymbol{\mu}_i - \mathbf{o}_w}{\|\boldsymbol{\mu}_i - \mathbf{o}_w\|_2}.$$

它与基元指向相机的方向相反。更换实现时必须核对这个符号：奇数阶球谐（spherical harmonics, SH）满足 $Y_{\ell m}(-\mathbf{d}) = (-1)^\ell Y_{\ell m}(\mathbf{d})$，所以反向后只保留原系数会改变颜色。

以实球谐的正交归一基底展开，各 RGB 通道独立具有一组系数：

$$\tilde{\mathbf{c}}_i(\mathbf{d}) = \tfrac{1}{2}\mathbf{1} + \sum_{\ell=0}^{L}\sum_{m=-\ell}^{\ell} \mathbf{h}_{i,\ell m} Y_{\ell m}(\mathbf{d}). \tag{4.5}$$

例如与原实现相容的一种排列为

$$Y_{00} = \frac{1}{\sqrt{4\pi}}, \qquad (Y_{1,-1}, Y_{10}, Y_{11}) = \sqrt{\frac{3}{4\pi}}(-d_y, d_z, -d_x).$$

零阶系数是可学习的常量颜色，不可在求和中遗漏。若初始颜色为 $\mathbf{c}_0$，则可取 $\mathbf{h}_{00} = (\mathbf{c}_0 - \frac{1}{2}\mathbf{1})/Y_{00}$，其余系数为零。这里的 $\frac{1}{2}$ 偏置、基底归一化、系数排列与方向必须作为一套编码约定保存；其他合法 SH 约定不能直接交换裸系数。

原始渲染实现对式 (4.5) 的负颜色作截断；它并不由 SH 展开自动保证颜色处于 $[0, 1]$。若额外使用 sigmoid 或双端截断，梯度及可表示的外观都会改变，应记录为实现选择。原版常见最高阶 $L = 3$，每通道共 $(L+1)^2 = 16$ 个系数，共 48 个浮点标量。高阶项能拟合复杂的视角外观，也可能把错误几何掩藏为方向变化；减少 SH 阶数既是压缩手段，也是需要独立验证的外观约束。

**直觉**

可把基元看作一个可旋转、可拉伸的局部响应。它的位置和椭球形状决定哪里能产生图像贡献，SH 决定从不同相机方向看它时是什么颜色。它不是从颜色中自动分离出了材质与光照；相机覆盖不足时，这些因素仍然纠缠。

## 4.3 从三维椭球到二维协方差

使用明确的世界到相机外参 $\boldsymbol{\mu}_c = R_{cw}\boldsymbol{\mu}_w + \mathbf{t}_{cw} = (x_c, y_c, z_c)^\top$。相机坐标扰动的协方差为 $\boldsymbol{\Sigma}_c = R_{cw}\boldsymbol{\Sigma}_w R_{cw}^\top$，平移不改变它。对针孔投影 $\pi$ 在均值处作一阶展开：

$$\pi(\boldsymbol{\mu}_c + \delta\mathbf{x}_c) \approx \pi(\boldsymbol{\mu}_c) + J\delta\mathbf{x}_c.$$

线性变换对协方差的作用给出

$$\boldsymbol{\Sigma}_{2D}^{(0)} = J R_{cw}\boldsymbol{\Sigma}_w R_{cw}^\top J^\top, \qquad \boldsymbol{\Sigma}_{2D} = \boldsymbol{\Sigma}_{2D}^{(0)} + \epsilon I_2. \tag{4.6}$$

上标 (0) 表示尚未加屏幕低通项。针孔相机的 Jacobian 为

$$J = \begin{bmatrix} f_x/z_c & 0 & -f_x x_c/z_c^2 \\ 0 & f_y/z_c & -f_y y_c/z_c^2 \end{bmatrix}, \qquad z_c > 0. \tag{4.7}$$

$J$ 的单位为像素/世界长度，$\boldsymbol{\Sigma}_{2D}$ 的单位为像素平方。主点影响投影中心，却不出现在 $J$ 中。该公式假设输入图像已去畸变且没有内参斜切；更一般的相机应使用其实际投影函数的导数。

### 4.3.1 一个可独立手算的投影

取 $R_{cw}=I$、$\mathbf{t}_{cw}=0$，令

$$\boldsymbol{\mu}_w=(1,2,4)^{\top},\quad \boldsymbol{\Sigma}_w=\mathrm{diag}(0.04,0.09,0.16),\quad f_x=400,\quad f_y=300.$$

这是教学合成输入，长度单位可自行选定；相应标准差为 0.2, 0.3, 0.4。投影中心为 $(c_x{+}100, c_y{+}150)$，有

$$J=\begin{bmatrix}100 & 0 & -25\\ 0 & 75 & -37.5\end{bmatrix},\qquad \boldsymbol{\Sigma}_{2D}^{(0)}=\begin{bmatrix}500 & 150\\ 150 & 731.25\end{bmatrix}. \tag{4.8}$$

例如右上角是 $(-25)(-37.5)\times 0.16=150$，说明沿深度方向的方差在斜视条件下会产生横纵相关性。若只删除三维矩阵的最后一行一列，就会漏掉这些项。矩阵的首个主子式为 $500>0$，行列式为 $343125>0$，因此正定。若选教学低通量 $\epsilon=0.3\ \mathrm{pixel}^2$，两个对角元素分别成为 500.3 和 731.55，非对角元素不变。

在固定三维协方差时，若将整个相机均值向量乘 2，即沿同一视线把距离加倍，则 $J$ 变为原来的一半，未低通协方差变为四分之一；若两个焦距加倍，则未低通协方差变为四倍。固定 $x_c, y_c$ 而只改变 $z_c$ 时，$J$ 的不同项缩放不同，不能直接沿用前一个比例。加入固定的 $\epsilon I$ 后，总协方差也不再严格按这些比例缩放。

### 4.3.2 正定不等于投影准确

当 $f_x, f_y>0$、$z_c>0$ 且三维协方差正定时，$J$ 具有行满秩。故对非零 $\mathbf{a}\in\mathbb{R}^2$，

$$\mathbf{a}^{\top}J\boldsymbol{\Sigma}_c J^{\top}\mathbf{a}=(J^{\top}\mathbf{a})^{\top}\boldsymbol{\Sigma}_c(J^{\top}\mathbf{a})>0.$$

在精确算术下，它本来就正定；$\epsilon I$ 还可改善浮点条件数，并设定最小屏幕足迹（footprint）。但最小足迹会改变渲染，不能把任意固定膨胀当作严格的像素积分或充分的抗混叠方案。

透视投影包含 $1/z$，只在均值邻域近似线性。以 $k$ 个标准差截断支持域时，深度半宽可估为 $k\sqrt{(\boldsymbol{\Sigma}_c)_{zz}}$；它应相对 $z_c$ 足够小，且显著支持域应位于近裁面之后。若高斯跨过近裁面、贴近相机或轴向过大，一阶投影可能严重失真，即使二维矩阵仍正定。原始实现还含视锥、视角范围与数值保护；复现时应逐项记录。增大 $\epsilon$ 不能修复错误外参、负深度或近裁面奇点。

第 12 章的脚本除上述闭式算例外，还用非恒等旋转和平移检查 $JR_{cw}$ 与中心有限差分的一致性，验证逆外参、正定性、尺度规律和中心近裁面拒绝。这些测试分别检查约定与实现，不等于证明大高斯的线性化误差可忽略。

## 4.4 屏幕响应、支持域与排序近似

投影中心为 $\boldsymbol{\mu}_{2D}=\pi(\boldsymbol{\mu}_c)$。在像素 $\mathbf{u}$ 处计算

$$g_i(\mathbf{u})=\exp\Big[-\tfrac{1}{2}\mathbf{r}_i^{\top}Q_i\mathbf{r}_i\Big],\quad \mathbf{r}_i=\mathbf{u}-\boldsymbol{\mu}_{i,2D},\quad Q_i=\boldsymbol{\Sigma}_{i,2D}^{-1},\quad a_i(\mathbf{u})=\alpha_i g_i(\mathbf{u}). \tag{4.9}$$

实作可以通过二维正定矩阵的解析逆或线性求解得到二次型，并检查行列式和有限性。这里同样使用未归一化高斯，不能擅自加入 $\det\boldsymbol{\Sigma}_{2D}$ 的归一化因子。

高斯有无限支持，光栅器必须截断。给定 Mahalanobis 半径 $k$，椭圆满足 $\mathbf{r}^{\top}Q\mathbf{r}\le k^2$，其轴对齐包围盒半宽分别为 $k\sqrt{\Sigma_{2D,11}}$ 与 $k\sqrt{\Sigma_{2D,22}}$；也可用最大特征值形成保守半径。包围盒覆盖的 tile 才进入候选列表。较小支持域节约计算，却会丢失尾部；不透明度阈值、包围盒取整和裁剪规则都是渲染定义的一部分。

原始 rasterizer 把屏幕分为 $16 \times 16$ 像素的 tile，将基元复制到覆盖的 tile，然后用 tile ID 与基元中心的相机深度组成键，一次进行 GPU radix sort[19]。同一 tile 中的像素共享这一中心排序，没有再按每个像素重新计算严格的相交次序。较小基元使此近似更合理；大而倾斜、相互穿插或深度接近的基元可能在局部需要不同顺序，产生跳变或遮挡错误。减小基元、改进排序和提高相机覆盖针对不同原因，不能只靠增加 SH 阶数修复。

## 4.5 前到后合成及其梯度

对覆盖像素且按当前规则排序的序列 $i = 1, \dots, n$，令

$$T_1 = 1, \quad T_{i+1} = T_i(1 - a_i), \qquad \widehat{\mathbf{C}}(\mathbf{u}) = \sum_{i=1}^{n} T_i a_i \mathbf{c}_i + T_{n+1}\mathbf{c}_{\mathrm{bg}}. \tag{4.10}$$

背景使用处理完最后一个基元后的 $T_{n+1}$。该式与离散体渲染同形，权重加背景仍为 1，但 $a_i$ 来自屏幕基函数，没有经过某个指定三维消光场的精确射线积分。合成所用颜色域应与第 1.4 节的声明一致；常见实现直接拟合输入编码域，不能把结果自动解释成线性辐射值。

把第 $i$ 个基元之后的全部贡献，包括背景，按进入该尾部的透射率归一化，记为 $\mathbf{C}_{>i}$。保持序列不变，可把包含第 $i$ 项的尾部写成 $a_i\mathbf{c}_i + (1 - a_i)\mathbf{C}_{>i}$，于是

$$\frac{\partial \widehat{\mathbf{C}}}{\partial \mathbf{c}_i} = T_i a_i I_3, \qquad \frac{\partial \widehat{\mathbf{C}}}{\partial a_i} = T_i(\mathbf{c}_i - \mathbf{C}_{>i}). \tag{4.11}$$

前方若已几乎不透明，$T_i$ 很小，后方基元就几乎收不到图像梯度；若该基元与其后方颜色相同，改变不透明度对图像也不敏感。这解释了只用 RGB 误差时容量和几何并非总能被唯一决定。

对未截断的式 (4.9)，在其他量固定时还有

$$\frac{\partial g_i}{\partial \boldsymbol{\mu}_{i,2D}} = g_i Q_i \mathbf{r}_i, \tag{4.12}$$

$$\frac{\partial g_i}{\partial \boldsymbol{\Sigma}_{i,2D}} = \tfrac{1}{2} g_i Q_i \mathbf{r}_i \mathbf{r}_i^{\top} Q_i, \tag{4.13}$$

$$\frac{\partial \boldsymbol{\Sigma}_i}{\partial \ell_{ij}} = 2 s_{ij}^2 \mathbf{r}_{ij} \mathbf{r}_{ij}^{\top}, \tag{4.14}$$

其中 $\mathbf{r}_{ij}$ 在最后一行特指 $R(\mathbf{q}_i)$ 的第 $j$ 列。该协方差导数中没有归一化概率密度才有的 $-\frac{1}{2}\Sigma^{-1}$ 项。对均值反传还要计入 $J$ 随深度变化和 SH 查询方向随均值变化的路径，不能仅保留屏幕中心的平移导数。四元数归一化的 Jacobian 则为

$$\frac{\partial \mathbf{q}}{\partial \widetilde{\mathbf{q}}} = \frac{I_4 - \mathbf{q}\mathbf{q}^{\top}}{\|\widetilde{\mathbf{q}}\|_2}.$$

上述链式求导给出自动微分与手写 CUDA 反传的核对依据；有限差分测试应避开排序交换、裁剪和截断阈值。

**适用条件与误区**

“可微光栅化”通常指固定可见列表与顺序后的分段可微。tile 归属、支持域取整、深度排序、阈值裁剪和提前终止都是离散选择，常规反传不对排序操作本身求连续导数。颜色截断、$a_i$ 上限和小透射率终止也会改变局部梯度。原实现对 alpha 设置小于 1 的上限，并跳过很小的贡献，以兼顾数值稳定与效率；复现应锁定这些规则。

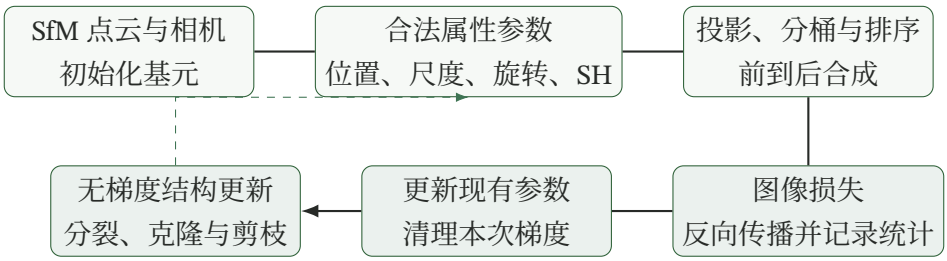



**图 4.1** 本章采用的安全训练时序。改变基元集合前完成当前参数更新，并显式维护优化器状态。

## 4.6 初始化、优化与容量控制

原始 3DGS 用已标定图像及 SfM 稀疏点云初始化均值，根据局部点间距设定初始尺度，由点颜色初始化零阶 SH，随后优化位置、尺度、旋转、不透明度与颜色 [19]。SfM 未覆盖的区域不一定能自行恢复，错误位姿也会被吸收到浮游基元和方向颜色中，因此初始化应与相机诊断一起检查。

与原论文的权重方向一致，基本损失写为

$$\mathcal{L} = (1-\lambda)\mathcal{L}_1 + \lambda\mathcal{L}_{\text{D-SSIM}}. \tag{4.15}$$

原论文报告使用 $\lambda = 0.2$；复现还要锁定 D-SSIM 的具体归一化、图像色域与裁边规则。

固定数量的基元并不总能同时解释大平面与细边缘。原方法在训练中积累可见基元的屏幕位置梯度，按观察次数归一化，作为欠拟合的启发式信号；大梯度说明图像希望改变当前投影，不等于已经确定了真实三维表面。在此基础上：

- 小尺度且高梯度的基元可以克隆，为未充分覆盖的局部增加容量；克隆后还需要后续优化使副本分工。
- 大尺度且高梯度的基元可以分裂为多个较小基元，以减少一团响应同时解释多个细节的困难。原方法用母基元的空间分布采样子位置并缩小尺度。
- 很低不透明度或异常大的基元可被剪枝；周期性降低不透明度是另一种训练调节，不能等同于基于完整评测路径证明“该基元无用”。

上述规则的阈值依赖场景尺度、图像分辨率、梯度归一化和训练阶段。过早剪枝可能丢失暂时被遮挡的细结构，持续分裂可能造成容量膨胀。应记录每次操作的数量、训练/验证误差、可见性与屏幕覆盖，检查增长是在补几何还是在拟合噪声。

### 4.6.1 结构变化必须与优化器状态协调

Adam 不只保存参数，还保存对应行的一、二阶矩。新增、删除或重排基元时，位置、尺度、四元数、SH、不透明度及其统计必须使用同一索引映射。如果反传后、`optimizer.step()` 前替换参数张量，刚得到的梯度可能留在旧张量上而不再被更新；形状相同也不能证明状态对应正确。

以下概念伪代码选取一种明确的安全顺序：先对旧集合完成当前更新，清空梯度，再在无梯度区修改集合。它不是要求所有官方实现必须逐行采用相同时序；采用其他时序的实现须显式迁移仍待使用的梯度及优化器状态。

**Listing 4.1** 安全维护参数集合的概念训练循环。统计每次累积，结构操作按间隔执行。

```
gaussians, optimizer = initialize_from_sfm(points, cameras)
for step in range(num_steps):
    camera = sample_training_camera()
    image, cache = rasterize(gaussians, camera)
    loss = photometric_loss(image, ground_truth[camera])
    loss.backward()
    # Detach statistics while the old parameter indexing is valid.
    stats.accumulate(cache.screen_gradient.detach(), cache.visible)
    optimizer.step()
    optimizer.zero_grad(set_to_none=True)

    if refinement_due(step):
        with no_grad():
            plan = propose_split_clone_prune(gaussians, stats)
            gaussians, index_map = apply_plan(gaussians, plan)
            optimizer = remap_parameters_and_adam_state(
                optimizer, gaussians, index_map,
                new_rows_policy="zero_moments")
            stats = remap_or_reset_stats(stats, index_map)
```

这里存活行保留相应 Adam 状态，删除行同步移除，新行按所选规则初始化，优化器参数组也要改指向新张量。伪代码对新行采用零动量；若具体方法要求继承或变换动量，应改写该策略并记录。分裂、克隆、重置不透明度、保存 checkpoint 和恢复训练都应有相同的状态一致性检查。验证时除了检查损失，还要断言所有属性与状态的行数一致、新参数实际进入优化器、保存后恢复能继续产生有限更新。

## 4.7 排序、像素覆盖与实际内存

对一张 $H \times W$ 图像，朴素 NeRF 的场查询成本可粗写为 $O(HWKC_{\text{field}})$，其中 $K$ 是每射线实际查询数，$C_{\text{field}}$ 包含编码及网络等查询工作；占据跳跃和分层采样会改变它。3DGS 则需分别计入投影、列表生成、排序和混合。

令 $N$ 为候选基元数，$M = \sum_i n_{\text{tile},i}$ 为基元 –tile 实例数，$P$ 为实际访问的基元 – 像素对数，则一种工作量分解为

$$O\big(N + M + \text{Sort}(M) + P + HW\big). \tag{4.16}$$

比较排序可写为 $O(M \log M)$；固定宽度键的 radix sort 则约为 $O(bM)$，$b$ 是遍历轮数，且包含多次显存读写。$P$ 受支持域、透明层数和提前终止影响，并不是高斯数 $N$ 的固定倍数。场景稀疏、足迹适中且列表带宽可承受时，这条 GPU 路径很有效；大量大足迹基元也可能使复制、排序和透明覆盖成为瓶颈。只写 $O(P)$ 会漏掉关键系统成本。

以最高阶 $L = 3$ 的普通表示为例，每个基元保存位置 3、log 尺度 3、四元数 4、不透明度 1 与 SH 48，合计 59 个标量。若均为 FP32，则 $10^6$ 个基元的裸属性为

$$10^6 \times 59 \times 4 = 236000000 \text{ bytes} = 236 \text{ MB} \approx 225.1 \text{ MiB}.$$

若梯度及 Adam 的两组矩也全按同样精度保存，仅这四组数组就约为 944 MB；这还不含排序键、tile 列表、图像、可见性缓存、反传缓存、分配器预留与 checkpoint 的其他数据。实际训练和渲染布局可以不同，因此这只是有条件的存储算例，不能替代显存测量。

磁盘码流、解码后属性、运行内存、训练峰值显存和渲染峰值显存是不同量。压缩后若仍把全部属性解码为 FP32，磁盘虽变小，渲染属性显存可能基本不变；若把若干属性改为按需生成，还需计算生成器、缓存与延迟。后续章节将把这些成本分别计入。

**本章回顾**

3DGS 用合法的旋转与正尺度组织几何，以 SH 表达方向颜色，通过一阶投影和分段可微的 alpha 合成形成训练图像。投影近似、中心深度排序、支持域截断和容量变化都有条件；理解这些条件，才知道量化或剪枝后的图像变化来自何处，以及应该测量哪个系统成本。

**练习与提示**

1. 把式 (4.8) 的均值改为 $(2,4,8)^{\mathsf{T}}$，保持三维协方差不变，重算未低通投影协方差。提示：先检查视线方向是否改变，再判断 $J$ 的比例。
2. 只有一个基元时，令 $\mathbf{c}_i = \mathbf{c}_{\mathrm{bg}}$，证明图像对 $a_i$ 的导数为零。这对仅凭颜色判断几何意味着什么？
3. 设两个有颜色差异的基元具有较高 alpha，交换顺序后图像一般会改变。写出两个顺序的差，说明低 alpha 或相近颜色为何会减小排序误差。提示：差为 $a_1 a_2(\mathbf{c}_1 - \mathbf{c}_2)$。
4. 将一百万个基元改为五百万个，估算 FP32 裸属性字节数，并列出不能仅按此倍数预测的三项开销。提示：排序实例数由足迹决定，不只由基元总数决定。

# 第 5 章　3DGS 的演化：尺度、结构、泛化与几何

标准 3DGS 把优化结果显式存为局部基元，但上一章的近似仍会在特定条件下暴露：像素足迹改变时会混叠，独立基元之间存在冗余，稀疏视角难以约束几何，静态场景假设也不能解释运动。本章按这些问题组织代表方法。选择方法时先核对输入与训练条件，再判断它改变了哪个表示或渲染环节；论文标题中的速度和压缩数字不构成相同条件下的比较。

## 5.1　抗混叠：限制可表示频率，再处理像素足迹

标准屏幕高斯只是连续图像的近似，而最终照片是有限像素足迹下的采样。相机拉远、焦距变化或图像缩小时，一个原本跨越多个像素的纹理可能小于一个像素；没有足够预滤波就会闪烁、出现摩尔纹或随尺度改变外观。训练图像本身的分辨率还限制了可辨认的三维细节：优化器不应无限缩小基元以解释没有被充分采样的高频。

Mip-Splatting 分别考虑这两个环节 [20]：用训练视角的采样信息约束三维高频，用二维 Mip 滤波处理屏幕采样，替换简单的固定膨胀。理解其动机的一种办法，是观察高斯响应与归一化高斯低通核的卷积。对协方差 $\Sigma$、峰值 $\alpha$ 的未归一化高斯，与协方差 $\Lambda$ 的归一化低通核卷积后，有

$$\Sigma' = \Sigma + \Lambda, \qquad \alpha' = \alpha\sqrt{\frac{\det\Sigma}{\det(\Sigma+\Lambda)}}. \tag{5.1}$$

该式在同一维度、正定协方差和上述核归一化条件下成立：足迹变宽，同时峰值下降，以保持积分。它说明仅增大协方差而保持峰值不变会增加响应面积。这里给出的是滤波原理，不把任意选择 $\Lambda$ 当作完整复现了 Mip-Splatting 的三维滤波量、可见性处理和二维实现。

抗混叠验证应固定同一场景和相机中心，在训练尺度附近及更小、更大的输出尺度比较结果，并正确更新内参；视频还应检查连续相机路径的时间闪烁。若真实几何错误、训练视角不足或曝光不一致，图像也会随视点变化，但这些现象不能全部归为采样混叠。滤波改善频率行为，并不能为未被观察到的表面创造监督。

## 5.2　结构化表示：共享什么，就约束什么

### 5.2.1　Scaffold-GS 的 anchor 与神经基元

标准 3DGS 分别存储每个基元的属性，局部相似性主要由优化间接形成。Scaffold-GS 用稀疏 anchor、局部特征和若干偏移来组织表示，再由共享的小网络根据方向与距离预测可见基元属性 [21]。示意写为

$$\begin{aligned} \boldsymbol{\mu}_{ij} &= \mathbf{a}_i + \mathbf{l}_i \odot \mathbf{o}_{ij}, \\ (\alpha_{ij}, \mathbf{c}_{ij}, \mathbf{s}_{ij}, \mathbf{q}_{ij}) &= g_\phi(\hat{\mathbf{f}}_i, \mathbf{d}_i, \rho_i)_j. \end{aligned} \tag{5.2}$$

其中 $\mathbf{a}_i$ 是 anchor，$\mathbf{o}_{ij}$ 是第 $j$ 个可学习偏移，$\mathbf{l}_i$ 为局部尺度因子，$\rho_i$ 表示相机距离；$\hat{\mathbf{f}}_i$ 由局部特征及多分辨率特征库融合得到。不同属性可用不同小网络预测，位置偏移并非全部由同一个网络临时生成，颜色也不必仍存成原始独立基元的完整 SH 列表。式中只保留机制所需变量，实际尺度组合和输出激活按官方实现锁定。

原方法的 anchor 来自对 SfM 点云的体素化，并通过基元梯度驱动的多分辨率增长与不透明度统计剪枝来调整。共享特征可减少重复属性，也把局部基元的误差耦合起来：一个 anchor 特征量化后，其全部子基元都可能改变。生成器还需要运算、权重和中间缓冲。因此“参数共享”应通过最终码流和端到端渲染测量验证，不能直接等同于某个固定压缩倍数。

Scaffold-GS 的视角条件生成通常仍是逐场景优化；共享的是该场景内部的生成机制，不能据此声称它已经具备跨场景零样本重建。结构化表示为层次细化、压缩和按需生成提供了接口，但原方法也不自动等于已经具有完整的 LOD 流式解码协议。若自行增添层级，应检查切换时的画质跳变、显存和加载成本。

### 5.2.2 分裂与克隆应服务于可诊断的误差

Feng 等的 *A New Split Algorithm for 3D Gaussian Splatting* 将分裂本身作为改进对象 [22]，提醒我们容量控制不是表示之外的小细节。比较分裂策略时，应固定初始化、优化步数、损失、剪枝规则与最终基元预算；同时观察大足迹、细边缘和浮游基元的变化。若新策略只是生成更多基元，画质提高与算法本身的容量分配效果就难以区分。

对后续压缩而言，训练期克隆与分裂决定了被编码对象的冗余结构。先用密度控制得到更合适的表示，和在固定表示上量化、剪枝，是不同的实验变量。二者可以联合设计，但应提供固定训练结果的后处理对照，避免把重训练收益全部记作编码收益。

## 5.3 跨场景泛化：从逐场景优化转向图像条件预测

前馈重建方法先在多个场景上学习从输入图像到三维表示的映射，再对新场景预测基元。MVSplat 是一个代表：输入带已知相机的稀疏多视图图像，利用多视图代价体推断几何，再预测用于渲染的高斯属性 [23]。这里的优势来自跨场景先验及输入视图之间的几何匹配；已知位姿、相机重叠和训练分布都是适用条件。

这条路线与逐场景 3DGS 的问题不同。后者的训练时间主要花在目标场景的参数优化，前者还需要事先的跨场景训练；只报告推理时间无法说明完整学习成本。稀疏输入之间没有重叠、反射导致匹配歧义、位姿不准或测试场景超出训练分布时，预测的不确定性会增加。可选的目标场景微调也应单独计时和报告数据使用情况。

公平评测至少包含场景级训练/验证/测试划分、输入视角选择规则、是否使用测试图像微调，以及推理网络权重的体积。部署多个场景时，共享网络可按明确规则摊销；只部署一个场景时，网络仍是真实依赖。由图像预测一个紧凑表示不必然构成压缩器：若没有定义输入对象、实际码流、解码器与重建失真，就不能把预测时间直接当成压缩解码时间。

## 5.4 几何一致性：2DGS 把基元贴到局部表面

仅有 RGB 监督时，空间上分散的半透明基元也可能形成相似图像。Huang 等的 2D Gaussian Splatting（2DGS）把基元改为嵌入三维空间的二维椭圆盘，显式描述切平面，再用透视一致的射线–盘求交进行渲染 [24]。其输入是带相机的图像与稀疏标定点云，原方法不以外部真实深度和法向作为普遍必需监督。

设中心为 $\mathbf{p}$，正交单位切向为 $\mathbf{t}_u, \mathbf{t}_v$，局部标准差为 $s_u, s_v > 0$，则盘上点为

$$\mathbf{P}(a,b) = \mathbf{p} + s_u\mathbf{t}_u a + s_v\mathbf{t}_v b, \qquad G(a,b) = e^{-(a^2+b^2)/2}. \tag{5.3}$$

法向 $\mathbf{n} = \mathbf{t}_u \times \mathbf{t}_v$。若将它形式上写成三维协方差，其法向特征值为零，是秩二的半正定矩阵；这不

是可以直接塞入式 (4.2) 求三维逆矩阵的普通 3D 高斯。

对单位方向射线 $\mathbf{r}(s) = \mathbf{o} + s\mathbf{d}$，在 $\mathbf{n}^\top \mathbf{d} \neq 0$ 时，射线与平面的交点满足

$$s_* = \frac{\mathbf{n}^\top(\mathbf{p} - \mathbf{o})}{\mathbf{n}^\top \mathbf{d}}, \qquad a = \frac{\mathbf{t}_u^\top(\mathbf{r}(s_*) - \mathbf{p})}{s_u}, \quad b = \frac{\mathbf{t}_v^\top(\mathbf{r}(s_*) - \mathbf{p})}{s_v}.$$

在局部 $(a, b)$ 处求高斯值，比只在中心线性化后投影整块盘更符合透视几何。该式是理解机制的几何推导；原实现采用适合 GPU 的齐次平面变换及显式求交，并含屏幕低通和数值保护。接近侧视时分母很小，求交不稳定；盘的支持域、近裁面和遮挡仍需处理。2DGS 也沿用中心深度的 tile 排序，因此不能声称透视正确求交已经消除了全部排序近似。

### 5.4.1 由自身渲染结果形成几何正则

2DGS 用图像损失加深度畸变与法向一致性正则：

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{rgb}} + \lambda_d \mathcal{L}_{\text{dist}} + \lambda_n \mathcal{L}_{\text{normal}}. \tag{5.4}$$

对同一像素的交点，令 $w_i = T_i a_i$ 为合成权重，$z_i$ 为统一约定下的交点深度，则深度畸变项的基本形式为

$$\mathcal{L}_{\text{dist}}(\mathbf{u}) = \sum_i \sum_j w_i w_j |z_i - z_j|.$$

它鼓励有效权重聚集在相近深度，抑制用多个分离透明层解释一个表面的情况；深度单位、距离参数化与损失权重须匹配具体实现，不能直接照搬另一数据尺度的 $\lambda_d$。

法向一致性比较朝向相机的盘法向 $\mathbf{n}_i$ 与渲染深度图反投影后估计的表面法向 $\mathbf{N}$，基本形式为

$$\mathcal{L}_{\text{normal}}(\mathbf{u}) = \sum_i w_i (1 - \mathbf{n}_i^\top \mathbf{N}), \qquad \mathbf{N} \propto \partial_u \mathbf{p}_s \times \partial_v \mathbf{p}_s.$$

$\mathbf{p}_s$ 是由相机与渲染深度得到的三维表面点，$\mathbf{N}$ 需归一化并与 $\mathbf{n}_i$ 统一方向。这里的监督来自模型自身的几何一致性，而非必须已有真实法向。有限差分在深度断裂、无效深度和轮廓处可能失稳，实际掩码、深度提取规则和梯度路径须锁定。

二维盘与正则更贴近不透明表面的假设，但透明介质、反射、头发和缺乏覆盖的薄结构仍会困难。若目标是网格提取，应报告深度/法向误差、表面精度与完整性或明确阈值的 F-score，并固定网格提取与清理流程；新视角 PSNR 提高并不独自证明几何更准。

## 5.5 容量分配的概率视角：MCMC 重定位与位置噪声

Kheradmand 等的 *3D Gaussian Splatting as Markov Chain Monte Carlo* 将高斯优化联系到随机梯度 Langevin 动力学，并重新设计探索与重定位 [25]。这不是泛指“随机出生和死亡”的通用算法，也不表示任意给全部属性加噪声都能得到正确的后验采样。

该方法主要向位置添加受不透明度及协方差调节的噪声；颜色、尺度和不透明度本身不采用同样的随机扰动。以所核查论文版本的记号改写，其位置探索项可表示为

$$\Delta\boldsymbol{\mu}_{\text{noise}} = \lambda_{\text{noise}} \eta \, \text{sigmoid}\big(k(\tau - \alpha)\big) \boldsymbol{\Sigma} \boldsymbol{\xi}, \qquad \boldsymbol{\xi} \sim \mathcal{N}(0, I_3). \tag{5.5}$$

它加在常规优化器的位置更新之外；$\eta$ 为位置学习率。低不透明度基元受到较强探索，高不透明度基元的噪声被抑制，协方差控制各向异性。这里按论文使用 $\boldsymbol{\Sigma}\boldsymbol{\xi}$，不能未经说明换成 $\boldsymbol{\Sigma}^{1/2}\boldsymbol{\xi}$ 后仍声称实现相同算法。实际方法采用 Adam、有限学习率和近似状态转移，其高质量重建结果不应被扩写成一般的精确采样或收敛保证。

重定位把低不透明度的“失活”基元移动到存活基元附近，按存活不透明度比例选择目标，并联合调整克隆后不透明度与协方差，以尽量减小图像响应变化。若把一个不透明度为 $\alpha$ 的基元替换成 $m$ 个同中心响应，要使中心合成不透明度保持不变，至少需满足

$$1-(1-\alpha')^m=\alpha, \qquad \alpha'=1-(1-\alpha)^{1/m}.$$

例如 $\alpha=0.75, m=2$ 时应取 $\alpha'=0.5$，而不是把 0.75 原样复制两次。但只匹配中心仍不够：离中心后 $1-(1-\alpha' g)^m$ 一般不等于 $\alpha g$。论文因此还修正协方差，使整体响应尽量接近，实际重定位仍是近似保持渲染，而非所有视角逐像素严格不变。方法还通过尺度与不透明度正则鼓励无用基元失活，以便容量重新分配。

这类操作必须维护 Adam 状态。所核查实现说明对目标基元与搬移基元采用不同的矩统计处理，以平衡停留和探索；不能把上一章通用伪代码的新行零动量策略机械套用到该方法。验证重定位时可先用一个高斯检查操作前后的响应曲线及积分，再比较固定预算下的训练稳定性、最终画质和覆盖范围。这样能区分机制是否复现与是否在特定数据上有效。

## 5.6 动态场景：时间加入以后，还要保持什么

静态表示将多帧中同一位置的不同外观视为不一致观测，运动物体会变成漂浮结构或模糊背景。动态高斯方法把时间加入表示，但“4DGS”是一个方法族名称，不能当作所有工作的同一实现。

Wu 等的 *4D Gaussian Splatting* 通过时空编码与形变模型，让规范空间中的高斯随时间改变 [26]。一种理解这种建模的示意是

$$\boldsymbol{\mu}_i(t)=\boldsymbol{\mu}_i^0+\Delta\boldsymbol{\mu}_i(t), \qquad \mathbf{s}_i(t)=\mathbf{s}_i^0\odot e^{\Delta\ell_i(t)},$$

旋转也按其实现采用合法参数更新；这两式只说明“规范属性加时间变化”的建模思路，并非替代原方法的完整网络和训练规则。表示可以共享静态属性，但还要保存或计算形变网络、时空特征和时间索引。

Luiten 等的 *Dynamic 3D Gaussians* 以多视角动态观测优化持久基元，使跨帧位置与方向相联系 [27]。持久性有助于跟踪和时序编辑，但基元标识不自动等于已经验证的物理材料点。多视角同步视频与单目移动相机视频对运动和几何的约束不同；阅读论文时应核对相机是否已知、是否同步、是否有静态背景及如何处理新出现与消失的表面。

动态压缩除了单帧属性，还要处理形变参数、时间残差、关键帧、可见性变化和随机访问。相邻帧 PSNR 高，并不能排除纹理闪烁或轨迹漂移。可补充遮挡掩码下的重投影一致性、已知轨迹误差、时间稳定性和随机时刻解码延迟；在没有参考运动时，应说明哪个指标只是自一致性，哪个得到外部验证。

SLAM 进一步把相机轨迹、在线地图和时间预算纳入闭环；语义高斯还会附加实例、语义或语言特征。它们改变了压缩对象和任务失真，不能仅测静态 RGB。本文到此只建立动态表示与系统的连接；动作条件动力学、闭环规划和机器人控制分别由世界模型与具身智能分册展开。

## 5.7 把方法选择写成条件与验证

表 5.1 以输入和机制组织选择依据；训练时间与 FPS 必须在固定场景、分辨率、硬件及实现下另行测量。它不对不同论文的结果作数值排名。

一次有解释力的消融只改变一个主要因素。例如研究 Mip 滤波，固定训练视图与密度控制；研究 anchor 压缩，分别计入共享网络和子基元的字节；研究几何正则，在相同网格流程下比较有无正

**表 5.1** 代表路线的适用条件与验证对象。前馈重建另含跨场景预训练成本，逐场景方法也可能采用额外先验。

| 路线 | 典型输入/训练条件 | 改变的机制 | 应重点验证 |
|---|---|---|---|
| NeRF/网格场 | 已标定多视图，逐场景优化 | 场查询与射线求积 | 采样误差、查询成本及几何 |
| 标准 3DGS | 已标定图像与 SfM 初始化 | 显式基元、中心排序合成 | 足迹、遮挡、列表带宽 |
| Mip-Splatting | 多视角及其采样尺度 | 三维频率约束、二维滤波 | 跨尺度质量与闪烁 |
| Scaffold-GS | 逐场景 anchor/特征优化 | 局部共享与视角属性预测 | anchor 耦合误差、生成成本 |
| MVSplat | 跨场景训练，稀疏已标定输入 | 多视图匹配及前馈高斯预测 | 场景泛化、输入覆盖与模型依赖 |
| 2DGS | 已标定图像与稀疏点云 | 平面基元、求交与几何正则 | 深度、法向、网格与侧视稳定性 |
| MCMC 3DGS | 逐场景优化，可研究不同初始化 | 位置探索、近似保持响应的重定位 | 固定预算下的覆盖与训练稳定性 |
| 动态高斯 | 时间观测；同步/相机条件依方法 | 形变或持久基元 | 时序、遮挡及时间访问成本 |

则；研究重定位，固定最大基元数并保存操作前后响应；研究动态编码，则在同一时间采样和参考帧下比较。所有对照都应锁定官方实现或自定义实现的版本，说明哪部分已经运行、哪部分仍是方案。

### 本章回顾

不同变体解决的瓶颈并不相同：滤波处理尺度采样，anchor 引入局部共享，前馈模型使用跨场景先验，二维基元及正则约束表面，重定位改善容量探索，动态表示引入时间关联。每一种改动也重新定义了后续要量化、剪枝和编码的属性，以及必须验证的失真。

### 练习与提示

1. 对一维未归一化高斯，验证式 (5.1) 的峰值修正使卷积前后积分相等。提示：积分为峰值乘 $\sqrt{2\pi\Sigma}$。
2. 如果一个共享 anchor 特征影响十个子基元，为何单独按每个高斯的颜色方差分配比特可能失效？提示：考察同一特征扰动对十个投影的相关影响。
3. 两个同中心克隆只按 $\alpha' = 1-\sqrt{1-\alpha}$ 修正不透明度、保持尺度不变时，比较 $g = 1$ 与 $g = 1/2$ 处的合成响应。提示：中心相等不代表整个函数相等。
4. 设计一个输入视角很少、但纹理丰富的场景，说明为何新视角 PSNR 与表面精度可能对 2DGS 和标准 3DGS 给出不同排序。列出所需的独立几何参考与网格提取条件。
5. 为动态高斯设计一组画质、运动与部署指标，并说明不能用没有参考运动的自一致性误差证明物理运动正确。

# 第二部分

# 压缩、率失眞与解码系统

# 第 6 章　从场景表示到率失真与部署预算

前几章回答了如何由图像拟合可渲染的场景。接下来要问：在目标设备和观看范围内，哪些信息必须保留，哪些信息可以少存、少算或稍后传输？3DGS 的显式属性使压缩容易入手，但“把浮点数变成整数”只覆盖了这个问题的一部分。读完本章，应能为一种压缩方案写出可核对的字节账本、质量目标与运行约束。

## 6.1　四种成本来自不同阶段

设训练后的场景为 $\mathcal{G} = \{g_i\}_{i=1}^{N}$，压缩码流为 $b$，解码结果为 $\widehat{\mathcal{G}} = \mathrm{Decode}(b)$。它们不是同一种数据结构，应区分表 6.1 中的成本。

**表 6.1**　压缩系统的成本边界。文件变小不自动意味着每项成本都降低。

| 阶段 | 需要计入 | 典型影响因素 |
|---|---|---|
| 磁盘与传输 | 完整码流、码本、模型、头部、块目录 | 位宽、分组、熵编码、共享模型的分摊规则 |
| 解码 | CPU/GPU 时间、临时内存、上传与首次可见延迟 | 上下文依赖、查表、网络推理、数据搬运 |
| 常驻内存 | 解码属性、索引、模型与常驻缓存 | 是否恢复为 FP32、是否即时查码本、属性布局 |
| 每帧渲染 | 投影、复制到 tile、排序、合成及峰值 workspace | 可见高斯数、屏幕覆盖、过绘制、分辨率 |

例如，8 bit 码字在磁盘上只占一个字节；若 viewer 在加载时把它展开成 FP32 属性，常驻属性内存仍由浮点布局决定。相反，即时查码本可减少常驻存储，却增加访存与解码指令。删除高斯通常同时减少属性和渲染工作，但若被删除的高斯原本就不可见，实际帧时间未必下降。训练 checkpoint 还可能保存梯度、优化器一阶与二阶矩以及密度控制状态，这些成本不可混入只为观看而设计的文件基线。

## 6.2　从 59 个标量建立可核对的基线

原始 3DGS 使用三维中心、三轴尺度、四元数、不透明度及每个颜色通道的实球谐系数 [19]。最高球谐阶为 $L$ 时，裸属性数量为

$$d(L) = 3 + 3 + 4 + 1 + 3(L+1)^2. \tag{6.1}$$

因此 $L = 3$ 时 $d = 59$，FP32 裸属性为 $236N$ byte。$N = 10^6$ 时是 236 000 000 byte，即 236 MB 或约 225.07 MiB。这里 1 MB 是 $10^6$ byte，1 MiB 是 $2^{20}$ byte。部分 PLY 布局另存三个占位法向分量，或者含有文本头与对齐，所以磁盘 PLY 不一定恰为 $236N$ byte。

编码账本应从实际文件出发：

$$\begin{aligned} B_{\mathrm{total}} = {} & B_{\mathrm{header}} + B_{\mathrm{geometry}} + B_{\mathrm{attributes}} + B_{\mathrm{indices}} \\ & + B_{\mathrm{codebook}} + B_{\mathrm{model}} + B_{\mathrm{directory}} + B_{\mathrm{padding}}. \end{aligned} \tag{6.2}$$

每项是否经过通用压缩、是否每场景重复保存，都应明确。共享解码器如果已安装，可报告“增量场景字节”；还应报告冷启动所需的完整模型字节，并说明分摊到多少场景。压缩比定义为同一协议下的 $B_{\mathrm{baseline}}/B_{\mathrm{total}}$，分母不能只统计最短的索引流。

**定义与约定**

**剪枝（pruning）**选择并删除基元或 anchor；**压实（compaction）**把保留元素、索引与状态重新排列成紧凑数组。只把门设为零未必减少实际处理量。**编码压缩（compression）**减少描述给定表示所需的字节。二者可以联合，但对帧时间、加载时延和显存的影响不同。3DGS.zip 综述 [28] 可作为路线索引，具体机制与数值须回到原论文及实现检查。

## 6.3 失真到底相对于谁

用原模型渲染图作参考，衡量的是**表示保真度**；用真实持出图像作参考，衡量的是最终重建质量。两者应分开写：

$$D_{\text{model}} = \mathbb{E}_{v\sim p_{\text{target}}}[d(R_v(\mathcal{G}), R_v(\hat{\mathcal{G}}))], \tag{6.3}$$

$$D_{\text{data}} = \mathbb{E}_{v\sim p_{\text{target}}}[d(I_v, R_v(\hat{\mathcal{G}}))]. \tag{6.4}$$

这里 $R_v$ 是固定渲染器，$I_v$ 是可用的真实图像。较小的 $D_{\text{model}}$ 不保证原场景本身准确；压缩恢复还可能偶然降低 $D_{\text{data}}$，不能因此声称编码无损。无真实图像的虚拟相机可用于检查轮廓连续性，但不提供新的真值证据。

视角分布 $p_{\text{target}}$ 是应用条件：相册浏览、前向行走和环绕观察的风险不同。训练集用于拟合，校准集用于统计量化范围或重要性，验证集用于选位宽、剪枝阈值与恢复轮数，最后测试集只用于最终报告。同一图像可在某个训练设定中兼作校准，但不能再称为独立测试。各章所说的“持出保护”指不参与梯度更新的验证视角；选定方案后仍需独立测试。

RGB 均方误差适于建立可复核基线，但 PSNR 会受像素范围、颜色空间、裁剪和背景影响。LPIPS 反映某种学习特征距离，也不等于几何可靠。若场景用于测量、导航或编辑，应增加深度、法向、表面距离或任务指标。几何真值不存在时，应把多视图一致性称为代理检查，而非准确度真值。

## 6.4 率失真目标与属性误差的传播

令 $S$ 是保留子集，$Q$ 是量化，$E$ 是无损编码；典型优化可写为

$$\begin{aligned}\min_{S,Q,E}\quad & D_{\text{data}}(\text{Decode}(E(Q(\mathcal{G}_S)))) + \lambda R_{\text{bits}} \\ & + \eta C_{\text{render}} + \gamma C_{\text{decode}},\end{aligned} \tag{6.5}$$

也可把内存、时延和失真写成硬约束。$R_{\text{bits}}$ 是完整码流位数；若用 byte 或每像素平均码率，$\lambda$ 的数值需随单位变换。这个目标是组织设计的语言，并不说明离散非凸问题已有全局最优算法。不同 $\lambda$ 得到的是一组工作点；最终要画率失真曲线并列出解码和显存约束，不能只选择一个有利工作点。

为什么不能最小化所有属性的欧氏 MSE ?把参数写成 $\theta$，在排序和可见性未突变的小邻域中，图像扰动近似为

$$\delta I_v \simeq J_v \delta\theta, \qquad D_{\text{local}} \simeq \delta\theta^{\top}\left(\mathbb{E}_v[J_v^{\top} W_v J_v]\right)\delta\theta. \tag{6.6}$$

$W_v$ 表示局部像素加权。位置、尺度、旋转与 SH 的量纲不同，$J_v$ 还随视角和遮挡变化，因此相同参数误差不产生相同图像误差。近轮廓、裁剪阈值、近远平面或深度排序交换处，一阶近似可能失效，应重新渲染验证，不能把局部二阶模型称为全局误差保证。

若暂时忽略属性间耦合，用 $h_g$ 表示第 $g$ 组的局部敏感度，范围宽度为 $r_g$，高分辨率均匀量化给出近似

$$D_g \simeq A_g 2^{-2b_g}, \qquad A_g = \frac{h_g r_g^2}{12}, \qquad \sum_g n_g b_g \le R_0. \tag{6.7}$$

这里 $n_g$ 是该组标量个数，$h_g$ 应与组内损失汇总方式一致。这个式子说明：多给 1 bit 在理想条件下使本组量化 MSE 约变为四分之一；它不说明任意属性都应多给同样位数。实际还受整数位宽、裁剪、分组元数据、混合编码和排序突变约束，可用小范围候选搜索验证该近似。

## 6.5 一个有头部与码本的算例

下面是教学文件设计，不是论文或真实场景压缩结果。设 $N = 10^6$，仍保存 $L = 3$ 的 59 个语义标量。表 6.2 给出一种未经熵编码的混合精度布局；四元数采用四分量量化并在解码后严格归一化。

**表 6.2** 教学混合精度布局。位宽仅用于核算，不是推荐默认值。

| 属性 | 分量数 | 每分量位数 | 每高斯位数 |
|---|---|---|---|
| 位置 | 3 | 16 | 48 |
| log 尺度 | 3 | 12 | 36 |
| 四元数 | 4 | 10 | 40 |
| 不透明度 | 1 | 8 | 8 |
| SH 的 $l = 0$ 项 | 3 | 10 | 30 |
| SH 的 $l = 1, 2, 3$ 项 | 45 | 6 | 270 |
| 合计 | 59 | — | 432 |

连续位打包的属性流是 54 000 000 byte。若每 256 个高斯一块，共 $\lceil 10^6/256 \rceil = 3907$ 块；每块保存 24 byte 的位置偏移/步长，共 93 768 byte。再假设其他全局范围共 448 byte、块目录每项 16 byte 共 62 512 byte，文件头 128 byte，则总计

$$B = 54\,000\,000 + 93\,768 + 448 + 62\,512 + 128 = 54\,156\,856 \text{ byte}.$$

相对于 236 000 000 byte 裸属性基线，压缩比约为 4.36。若解码后完全展开为原 FP32 属性，则常驻裸属性仍需约 236 MB；此时“约 54 MB 的文件”不能写成“约 54 MB 显存”。真实协议还需按实际对齐、尾块长度及校验和修正这个账本。

再考虑只把每高斯 48 维 SH 向量替换为 $M = 256$ 的 FP32 码字。固定长索引是 1 000 000 byte，码本是 $256 \times 48 \times 4 = 49\,152$ byte，二者合计约 1.05 MB；但位置等 11 个 FP32 标量仍需 44 MB。整个场景约 45.05 MB，绝非把全部 236 MB 都除以 192。向量量化是否能保持画质，由数据和渲染验证决定；这里只核算了存储。

## 6.6 可比较的压缩实验

最小报告应固定同一个训练结果、同一组测试相机、背景与渲染器，依次比较仅剪枝、仅量化、仅熵编码和联合方法。仅熵编码属于对离散符号的无损重编码，若解码符号与基线相同，图像应在同渲染数值条件下相同；若图像变化，应先排查协议、反量化与渲染实现。

同时报告原始和压缩完整字节、PSNR/SSIM/LPIPS、解码时间、加载到首帧的时间、解码后常驻内存、峰值显存和稳定渲染帧时间。预热和 GPU 同步方式要一致。首帧体验可分成传输、熵解码、属性重建、上传和渲染；流水并行时不能简单把各阶段单独测得的峰值相加当作端到端时延。

**本章回顾**

率失真比较先明确参考图像、观看分布和字节边界，再讨论算法。基元数量、文件大小、解码显存与 FPS 是互相关联但不等价的结果。局部敏感度能指导位宽，最终结论仍需由独立视角和完整码流验证。

**练习与提示**

1. 按式 (6.1) 算出 $L = 0, 1, 2$ 的属性数；说明降低 SH 阶与量化 SH 为什么是不同的失真来源。提示：比较可表达的函数空间。
2. 若教学混合精度码流每块再增加 32 byte 校验和，更新总字节与压缩比。若索引采用每项 `uint16` 而非真正的 10 bit 打包，哪些预算会变化？
3. 构造两个场景表示：第一种文件更小但加载更慢，第二种高斯更少但帧时间不变。分别指出可能占主导的成本。
4. 设两组属性的 $A_g$ 相差 16 倍、$n_g$ 相同，在式 (6.7) 的近似下，最优位宽大约相差多少？提示：比较每增加一位带来的失真下降。

# 第 7 章　高斯属性量化：离散坐标、码本与约束

量化把连续属性映射为有限符号。它有三个不同层面：在哪个坐标域量化、如何选择离散码、如何把码写进文件。前两者影响渲染失真，最后一项决定实际字节与解码成本。本章先建立标量量化，再处理高斯属性的几何约束，最后讨论向量量化、混合精度与训练协议。

## 7.1　标量量化：把零点、偏移与裁剪说清楚

对整数范围 $q_{\min}, \dots, q_{\max}$，整数零点形式是

$$q = \operatorname{clip}\left(\operatorname{round}(x/\Delta) + z, q_{\min}, q_{\max}\right), \tag{7.1}$$

$$\hat{x} = \Delta(q - z), \qquad \Delta > 0, \quad z \in \mathbb{Z}. \tag{7.2}$$

$z$ 是实数零在整数域的位置。选择无符号 $b$ bit 时，$q_{\min} = 0$、$q_{\max} = 2^b - 1$；有符号对称量化常用 $[-q_{\max}, q_{\max}]$ 并令 $z = 0$，其有效等级数未必等于整数类型全部可用值的数量。

另一种同样合法、但记号不同的端点形式是

$$\Delta = \frac{x_{\max} - x_{\min}}{K - 1}, \quad q = \operatorname{clip}\left(\operatorname{round}\frac{x - x_{\min}}{\Delta}, 0, K - 1\right), \quad \hat{x} = x_{\min} + \Delta q. \tag{7.3}$$

这里 $x_{\min}$ 是实数偏移，不要求 $-x_{\min}/\Delta$ 是整数；它与式 (7.2) 的零点约束不可直接混用。协议必须记录端点、整数零点或实数偏移中的实际一种，并约定舍入规则、字节序和位序。常数块 $x_{\max} = x_{\min}$ 应用常量标志解码，不能计算零步长再除零。

无裁剪时，最近邻量化有 $|x - \hat{x}| \le \Delta/2$。进一步假设误差在量化格内近似均匀，才有

$$\mathbb{E}[e] \simeq 0, \qquad \mathbb{E}[e^2] \simeq \frac{1}{\Delta}\int_{-\Delta/2}^{\Delta/2} e^2\, de = \frac{\Delta^2}{12}. \tag{7.4}$$

低位宽、尖峰分布、与格点相关的参数及范围裁剪都可能破坏均匀噪声近似。裁剪误差还包括范围外的长尾；用百分位代替 min–max 可以改善多数值的分辨率，但必须统计被裁剪的高斯是否位于关键轮廓。

## 7.2　分组粒度与范围选择

全张量共享范围便宜，但一个离群点便可能拉大所有值的步长。按属性维度、SH 阶、空间块或 anchor 分组可以减小范围，同时引入更多偏移、尺度和组索引。设每组需 $m$ byte 元数据、每组 $n$ 个高斯，则分组本身的平均成本为 $m/n$ byte/高斯；不能只根据属性 MSE 选择最细粒度。

范围可以由 min–max、百分位、属性 MSE、渲染损失或联合率失真决定。应分别统计量化网格误差和裁剪误差，保存精确范围、分组顺序与异常值处理规则。对明显的离群高斯，可采用“常规块低位宽加少量逃逸符号”的设计：逃逸符号指向高精度残差流，其索引和长度同样计费。这比为了极少数异常值让整幅场景增加位宽更灵活。

## 7.3　位置：场景尺度、局部坐标与轮廓

若对宽度 $W$ 的轴使用 $b$ bit 端点量化，步长是 $W/(2^b - 1)$。场景坐标放大 100 倍，若不改变位宽和归一化，绝对误差也放大 100 倍。因此保存场景归一化变换本身就是编码协议的一部分。

对相机坐标 $(x, y, z)$，投影 $u = f_x x/z + c_x$ 的一阶扰动为

$$\delta u \simeq \frac{f_x}{z}\delta x - \frac{f_x x}{z^2}\delta z. \tag{7.5}$$

例如 $f_x = 800$ pixel、$z = 2$ m，横向位置误差 1 mm 在忽略深度误差时约对应 0.4 pixel；同样误差在近处更敏感。这是教学局部估算，穿过遮挡边界时还可能改变合成顺序。

局部位置可写成

$$\xi_{ij} = \frac{\mu_{ij} - a_i}{r_i}, \qquad \widehat{\mu}_{ij} = \hat{a}_i + \hat{r}_i\,\hat{\xi}_{ij}. \tag{7.6}$$

这里 $a_i$ 是 anchor 或块原点，$r_i > 0$ 是局部尺度。即使残差很精确，anchor 与尺度的误差也会传递给所有子高斯；一阶展开为 $\delta\mu \simeq \delta a + \xi\,\delta r + r\,\delta\xi$。因此 anchor 不是免费且无误差的参照。Scaffold-GS 提供了可利用的 anchor 结构 [21]，但局部坐标也可为普通高斯集合额外建立，需计入分块和索引成本。

检查位置量化时，先在统一世界单位中测最大误差，再比较投影轮廓、近处细杆和深度相近的前后层。对于无界场景，不宜用单个巨大包围盒压缩全部位置；可分块、分层或为远处/离群点单列编码方式，并把坐标恢复写成可逆协议。

## 7.4 尺度与旋转：合法性不等于渲染等价

### 7.4.1 对数尺度控制相对误差

若三轴尺度 $s_j > 0$，取 $\ell_j = \log s_j$，解码 $\hat{s}_j = \exp(\hat{\ell}_j)$。它严格保持正值，且

$$\frac{\hat{s}_j}{s_j} = e^{\delta\ell_j}, \qquad \frac{\hat{s}_j - s_j}{s_j} \simeq \delta\ell_j. \tag{7.7}$$

对数域误差 0.01 大致对应 1% 尺度相对误差。实现应规定有限的 $\ell$ 范围、指数计算精度与最小尺度。若原模型已有 log 尺度，不要重复取对数；对 $\log(s + \epsilon)$ 量化后再减 $\epsilon$ 可能恢复为非正值，不能拿它代替合法性检查。

协方差为 $\Sigma = R\,\mathrm{diag}(s_1^2, s_2^2, s_3^2)R^\top$。排序尺度轴时，必须同步变换旋转基，且保持 $\det R = 1$；只把三个尺度从小到大排序会改变协方差。协方差的等价参数化可有利于编码平滑，但必须验证恢复前后矩阵一致。

### 7.4.2 四元数的双覆盖与归一化

单位四元数 $q$ 和 $-q$ 表示同一旋转。直接做欧氏聚类前，应统一符号或使用 $\min(\|q-c\|_2, \|q+c\|_2)$ 这样的符号不变距离。固定 $q_w \geq 0$ 是一种约定，但在接近 180° 的边界附近仍有不连续性。旋转误差可用

$$d_R(q, \hat{q}) = 2\arccos\bigl(\min(1, |q^\top \hat{q}|)\bigr) \tag{7.8}$$

来检查，前提是二者已归一化。

逐分量解码得到 $\tilde{q}$ 后，应执行

$$\hat{q} = \begin{cases} \tilde{q}/\|\tilde{q}\|_2, & \|\tilde{q}\|_2 > \varepsilon, \\ q_{\text{fallback}}, & \text{否则}, \end{cases} \tag{7.9}$$

并令回退四元数严格为单位四元数；损坏码流可改为拒绝解码。分母加 $\epsilon$ 的写法只使范数接近一，不是严格归一化。

一种紧凑设计是选最大绝对值分量 $j$，统一其符号为正，保存 2 bit 的 $j$ 与另外三个分量。其余分量绝对值不超过 $1/\sqrt{2}$，解码以 $q_j = \sqrt{\max(0, 1 - \sum_{k \neq j} \hat{q}_k^2)}$ 恢复，再归一化。取最大值的规则、并列处理、三个分量的顺序与截断规则都需固定；量化后的平方和可能略大于一，不能省略边界处理。

局部旋转也可写成 $R = R_0 \exp([\omega]_\times)$，在足够小的邻域量化三维切向量 $\omega$，但需保存或推导 $R_0$，并处理接近 $\pi$ 的分支。6D 旋转表示则常通过正交化恢复旋转矩阵，与指数映射不是同一操作。更换参数化后，必须比较投影协方差；各向同性高斯对旋转不敏感，极细长高斯却可能放大很小角度误差。

## 7.5 不透明度与球谐：选择域而非背诵位宽

### 7.5.1 概率域与 logit 域各自分配什么精度

用 $o \in [0, 1]$ 表示高斯的不透明度振幅，用 $a_i(u)$ 表示乘上屏幕高斯后的逐像素 alpha，避免混淆。常见训练参数为 $t$，$o = \mathrm{sigmoid}(t)$。概率域均匀量化直接控制绝对误差 $|\delta o| \le \Delta_o/2$；logit 域量化通过

$$\frac{do}{dt} = o(1 - o), \qquad |\delta o| \lesssim o(1 - o)|\delta t| \tag{7.10}$$

分配误差。同一步长的 logit 量化在接近 0 或 1 处产生更小的绝对概率变化，在 $o = 0.5$ 附近变化最大；全局还有 $|\delta o| \le |\delta t|/4$ 的 Lipschitz 上界。因而不能普遍声称 logit 域给遮挡边界更多精度。应结合目标视角上的透射率和后景颜色选择域与步长。

概率恰为 0 或 1 时 logit 无穷大。可以用特殊端点符号，或先裁剪到 $[\epsilon, 1 - \epsilon]$，但后一种处理已经改变原表示，需计入失真。若用有限 logit 范围，报告范围、裁剪比例和 sigmoid 数值精度。量化与剪枝阈值应在同一语义域中比较；渲染器自己的 alpha 截断、跳过阈值和早停阈值也应固定。

### 7.5.2 球谐的阶数、符号与方向约定

设线性颜色在不作截断时写成

$$c(d) = c_{\text{bias}} + \sum_{l=0}^{L} \sum_{m=-l}^{l} f_{lm} Y_{lm}(d), \qquad \delta c(d) = \sum_{l,m} \delta f_{lm} Y_{lm}(d). \tag{7.11}$$

同一组数字只有在实球谐归一化、$m$ 次序、方向是“相机到点”还是“点到相机”、系数布局和颜色偏移一致时才有相同含义。反转方向会使奇数阶项变号；把有正有负的 SH 系数当无符号幅值存储会丢失函数，不能靠最后裁剪颜色恢复。

若基函数在球面正交归一，方向按整个球面均匀分布，则对单通道有

$$\mathbb{E}_d[(\delta c(d))^2] = \frac{1}{4\pi} \sum_{l,m} (\delta f_{lm})^2. \tag{7.12}$$

实际相机并不均匀，截断、遮挡和颜色空间还会改变这个关系。按 SH 阶分配位宽是有用基线，但“高阶总不重要”是错误的：窄高光和斜视细节往往更依赖高阶项。先单独测试截阶，再单独测试系数量化；通过蒸馏恢复截阶的收益应单独报告。LightGaussian 将这类组件与剪枝、量化组合 [29]，不能把其联合结果全部归因于某一项。

## 7.6 向量量化、残差码与真实索引成本

标量量化忽略相关性。向量量化（VQ）为 $d$ 维属性学习码本 $\mathcal{C} = \{c_k\}_{k=1}^{M}$，编码

$$k^*(x) = \arg\min_{1 \le k \le M} \|x - c_k\|_2^2, \qquad \hat{x} = c_{k^*(x)}. \tag{7.13}$$

属性单位不同，直接把位置与 SH 拼接做欧氏距离可能让某组主导，可归一化、分码本，或用渲染加权度量。白化/归一化参数也必须进入解码协议；球面旋转不能按普通未归一化向量任意聚类。

固定长索引需 $\lceil \log_2 M \rceil$ bit/向量，码本若每分量 $b_c$ bit，则理想位预算为

$$R_{\mathrm{VQ}} = N\lceil \log_2 M \rceil + Mdb_c + R_{\mathrm{metadata}}. \tag{7.14}$$

例如 $M = 256, d = 48, b_c = 32$，码本为 49 152 byte。若只编码 $N = 1000$ 个 SH 向量，索引虽只需 1000 byte，码本却不能忽略；在 $N = 10^6$ 时才被充分摊薄。$M = 300$ 需 9 bit 固定长索引；用 `uint16` 存储而不打包时实际是 16 bit，不能仍按 9 bit 报告。

产品量化将一个向量拆成若干子空间，降低最近邻搜索成本；残差量化则依次编码尚未解释的误差：

$$r^{(0)} = x, \quad k_j = Q_j(r^{(j-1)}), \quad r^{(j)} = r^{(j-1)} - c_{k_j}^{(j)}, \quad \hat{x}^{(J)} = \sum_{j=1}^{J} c_{k_j}^{(j)}. \tag{7.15}$$

若 48 维向量用 4 个 $M = 256$ 的完整维度残差码本，索引流是每向量 4 byte、码本共 196 608 byte；这不同于把 48 维分成四个 12 维子空间，其总代码本只有 49 152 byte。残差层数增加不会自动形成可用渐进码流：每个前缀还需训练/验证，其依赖和合法属性恢复规则见下一章之后的学习压缩讨论。

码本训练可用 k-means、软分配或基于渲染损失的联合学习。硬索引不可微，常见直通估计器（STE）写成

$$\hat{x}_{\mathrm{st}} = x + \mathrm{stopgrad}(Q(x) - x), \qquad \frac{\partial \hat{x}_{\mathrm{st}}}{\partial x} \approx I. \tag{7.16}$$

它是梯度代理，不是离散算子的真实导数。Noise-Substituted VQ[30] 研究用噪声替代训练时硬量化；噪声训练完成后仍需以实际最近邻索引、实际码流和独立视角测试，不能把连续噪声损失直接当作部署损失。

## 7.7 PTQ、QAT 与混合精度的实验顺序

后训练量化（PTQ）固定训练结果，估计范围、学习码本并编码；它不必重新优化全部场景，但学习码本本身仍有计算成本。量化感知训练（QAT）在优化中插入伪量化或离散代理，使属性适应有限精度；它不保证比所有 PTQ 方案更好，且必须报告恢复步数、训练时间与需要的图像。

建议先固定场景做逐属性 PTQ 扫描，发现敏感组，再做混合位宽搜索。把每种属性先独立量化，能区分位置轮廓、尺度模糊、旋转形状、不透明度遮挡与 SH 高光误差；联合量化时再观察交互。分组范围、码本容量和异常值流要在验证集上选，最终测试视角不参与选择。若 QAT 使用原训练相机优化，在验证相机上早停，最后保留一组未使用的相机测试。

**Listing 7.1** 量化评测应经过实际文件往返，而非直接渲染伪量化张量。

```
scene = load_frozen_scene(checkpoint)
stats = calibrate_ranges(scene, calibration_views)
for policy in candidate_policies:
    codes, side_info = quantize(scene, stats, policy)
    path = write_complete_stream(codes, side_info)
    decoded = decode_from_file(path)  # no encoder tensors
    assert_valid_scales_rotations(decoded)
    report = evaluate(decoded, validation_views)
    report.bytes = file_size(path)
policy = select_with_budgets(all_reports)
evaluate(decode_from_file(policy.path), final_test_views)
```

解码器必须只依赖文件和明示的共享资源。若测试时还在读取编码器内存中的浮点中心、原始码本或归一化系数，测试并未完成。还应测试所有值相同、全透明、极小尺度、四元数符号翻转、码本空簇、最后不满字节与尾块不满组等边界。

## 7.8 混合精度如何与实现相接

混合精度是对不同属性或块使用不同位宽，而不是固定的“位置最多、旋转最少”排序。可以估计每组增添一位所降低的验证失真 $\Delta D_g$，除以新增完整字节 $\Delta B_g$，再按收益选择候选；但这是启发式，分组、裁剪与剪枝联动时需重新测量。

只有编码器真正写出位流，非标准位宽才带来相应字节节省。GPU 若仅支持 8/16/32 bit 向量加载，6 bit 码流可能先在 CPU 展开，或在 GPU 用位操作拆包。两种实现均合理，但要报告解码时间、带宽和运行布局。VQ 也是如此：预先展开更快地接入标准 viewer；保持索引常驻则需自定义查表渲染器，其 kernel、缓存和随机访存成本应单独测量。

**本章回顾**

量化先保持语义与约束，再优化码率。位置需要一致的单位与局部参照，尺度需要正值，旋转需要合法 $SO(3)$，不透明度需要明确端点，SH 需要固定基函数与符号。标量量化、VQ、残差码和熵编码各有收益来源；完整码流往返及持出视角验证把这些选择变成可检验结论。

**练习与提示**

1. 对 $x \in [-1, 3]$ 使用 $K = 4$ 的端点量化，列出四个恢复值，并求 $x = 0$ 的恢复值。它是否能表示精确零？再设计一个整数零点方案，比较差异。
2. 推导位置局部残差的误差传播式。若 256 个子高斯共享一个量化 anchor，为什么它们的误差不独立？
3. 对 $q = (1/2, 1/2, 1/2, 1/2)$ 和 $-q$ 比较原始欧氏距离与式 (7.8)。构造一个量化后范数接近零的输入，说明回退策略为何必须明确。
4. 在 $o = 0.01, 0.5, 0.99$ 处分别计算 $o(1 - o)$，据此解释 logit 域固定误差如何映射到概率误差。不要据此跳过真实渲染比较。
5. 对 $N = 10^4, d = 48, M = 300$ 的 FP32 VQ 码本，分别按 9 bit 打包与 `uint16` 索引计算总大小，暂不计头部。提示：包含码本 $300 \times 48 \times 4$ byte。
6. 若一个高光仅在验证集的两个斜视角出现，如何避免在平均 PSNR 基本相同的两个位宽方案中选错？提示：分区域与分视角报告，并保持独立测试。

# 第 8 章　剪枝与压实：从贡献分数到删除风险

量化保留基元而减少每个属性的精度；剪枝直接改变表示的组成。它既可能减少文件，也可能减少投影、排序和像素合成工作。然而高斯不是互不干扰的零件：前景会改变后景的透射率，多个小透明度也能累积成明显遮挡。本章从这个耦合出发建立删除分数、渐进流程与验证边界。

## 8.1　密度控制与部署剪枝解决不同阶段的问题

训练时的克隆、分裂和剪枝共同控制容量：

$$N_{t+1} = N_t + \Delta N_{\text{densify}}(t) - \Delta N_{\text{prune}}(t). \tag{8.1}$$

密度控制尝试把表示能力放到欠拟合区域，也清除低不透明度或异常基元；部署剪枝则通常从已训练结果出发，在给定质量和设备约束下减小场景。二者可以联合学习，但实验不能把更长训练、更强分裂策略和最后的压缩混为一个收益。

在 densification 尚未稳定时过度删除，会使模型缺少表达细结构的机会。另一方面，训练到收敛后再大量剪枝，也可能需要额外恢复计算。因此至少保存未压缩 checkpoint、训练预算和最终高斯数，再分别比较不恢复与固定步数恢复的剪枝结果。LightGaussian 的剪枝、恢复、SH 蒸馏与 VQ[29] 提供了组合路线，但应对组件逐个消融。

## 8.2　删除一个高斯究竟改变什么

考虑某像素的理想前到后 alpha 合成，固定剩余高斯的属性和顺序。将第 $i$ 个高斯前的累计颜色记为 $C_{<i}$、透射率记为 $T_i$；把后方所有高斯连同背景从透射率一开始合成的颜色记为 $C_{>i}$，则

$$C = C_{<i} + T_i\big[a_i c_i + (1 - a_i) C_{>i}\big], \tag{8.2}$$

$$C^{\setminus i} = C_{<i} + T_i C_{>i}, \tag{8.3}$$

$$C - C^{\setminus i} = T_i a_i (c_i - C_{>i}). \tag{8.4}$$

式 (8.4) 在固定排序、完整数学合成下精确成立。实际渲染器若有 alpha 跳过或透射率早停，删除可能改变执行路径，应以真实实现复查。关键结论是：重要性不仅来自 $T_i a_i c_i$，还来自被它遮挡的后景。

**适用条件与误区**

黑色遮挡物不是“没有颜色贡献”。令前方透射率为 1、$a_i = 0.8$、$c_i = 0$，后景为白色 1，则原颜色是 0.2，删除后是 1，差为 0.8。用 $T_i a_i \|c_i\|$ 打分却得到零，因而不能把该分数当删除误差。

定义真正的删除损失增量为

$$\Delta_i = \mathcal{L}(\mathcal{G} \setminus \{i\}) - \mathcal{L}(\mathcal{G}). \tag{8.5}$$

正值意味着删除变差，负值意味着原高斯可能在当前指标与视角上有害。若优化已把训练图像拟合得很好，一阶训练梯度可以接近零，但删除造成的有限扰动仍可很大；“梯度小”不是可删证明。

可引入连续保留门 $m_i \in [0, 1]$，令 $a_i$ 被 $m_i a_i$ 替代。在 $m_i = 1$ 附近，删除对应 $\delta m_i = -1$，二阶代理为

$$\Delta_i \approx -\frac{\partial \mathcal{L}}{\partial m_i} + \frac{1}{2}\frac{\partial^2 \mathcal{L}}{\partial m_i^2}. \tag{8.6}$$

因为扰动幅度为一、并不小，这只是候选排序依据。一次删除多个高斯还引入 Hessian 交叉项，以及透射率和可见性的变化；逐个分数相加通常不是子集删除损失。

## 8.3 便宜分数的作用与盲区

可以先用便宜代理筛选候选，再用少量真实渲染评估批量删除。表 8.1 列出常用分数，它们表达不同的信号，不宜混用名称。

**表 8.1** 重要性代理是筛选工具，不是无条件误差保证。

| 信号 | 可提供的信息 | 主要盲区 |
|---|---|---|
| 不透明度 $o_i$ | 快速发现近透明基元 | 不含可见性、覆盖面积、累积遮挡 |
| 透射率加权覆盖 | 估计目标相机实际访问的像素权重 | 可能偏爱大 footprint；未覆盖背面未知 |
| 训练梯度/二阶代理 | 当前损失对参数或门的敏感度 | 收敛点小梯度、有限删除、近似开销 |
| 尺度或体积 | 发现异常大、扁或孤立的基元 | 几何大小不等于图像必要性 |
| 局部聚类与重叠 | 识别相近位置/形状/外观的候选 | 遮挡顺序和颜色差可能破坏替代 |
| 任务/边界保护 | 保留细杆、文字、深度断裂和目标 | 依赖检测器与视角覆盖，仍需图像复核 |

以固定校准相机集合 $\mathcal{V}_{\text{cal}}$ 统计加权覆盖，可定义

$$s_i^{\text{cover}} = \frac{1}{|\mathcal{V}_{\text{cal}}|} \sum_{v \in \mathcal{V}_{\text{cal}}} \sum_{u \in \mathcal{P}_i(v)} T_i(u) a_i(u). \tag{8.7}$$

它没有把颜色范数误认为全部影响，但仍不是式 (8.5)。若能获取后景，$T_i a_i \|c_i - C_{>i}\|$ 更接近单次删除的像素变化；然而生成这些统计本身有渲染和显存成本。应以统一实现比较“分数更准”是否值得其计算代价。

一个低不透明度高斯对单像素影响可能小，许多这样的高斯却能形成大的总不透明度 $1-\prod_i(1-a_i)$。例如十层 $a_i = 0.1$ 的累计不透明度约为 0.651。批量按阈值全部删除，会产生远大于单个分数的变化。因此每轮小步删除并重新测量比一次性 top-$K$ 更容易控制风险。

## 8.4 重叠、冗余与理论保证的边界

若将高斯看作归一化函数 $\phi_i(x) = \mathcal{N}(x; \mu_i, \Sigma_i)$，其内积有解析式

$$\int \phi_i(x)\phi_j(x)\,dx = \mathcal{N}(\mu_i; \mu_j, \Sigma_i + \Sigma_j). \tag{8.8}$$

这可建立局部 Gram 矩阵，发现位置与形状高度相关的基函数。若使用的是未归一化峰值高斯，必须加上对应幅值与行列式因子，不能直接套用右式。更关键的是，3DGS 的最终图像并非这些三维函数的线性求和：投影、SH、alpha 乘积和深度顺序都参与成像。

最简单的反例是两个重叠白色基元，某像素上各有 $a = 0.5$，背景黑色。合成颜色为 0.75；删除一个后是 0.5，尽管它们的基函数相同。尝试提高剩余不透明度，也未必在所有像素同时保持原 footprint：$1-(1-oG)^2$ 包含 $G^2$ 项，通常不能写成某个常数 $o'$ 乘同一 $G$。功能相似是提出替代候选的理由，而非已经证明可无损合并。

学习式与统计式剪枝包括 LP-3DGS[31]、Trimming the Fat[32] 及置信分布方案 [33]。核验于 2026-10-21 并于 22 日复查的近年预印本还提供三个不同观察窗口：REFINE 用解析近似的渲染感知 Hessian 场做代理 [34]；QIRF 用局部非正交基与响应子空间选择候选 [35]；GETA-3DGS 探索结构化删除与混合精度联合分配 [36]。它们不能仅凭标题视为同一算法，本文也未独立复现其性能。Jo 等的聚类方法则在离线建立空间簇，并按运行时当前视角提前剔除不可见基元 [37]；这是视角相关的加速剔除，并不永久删除场景，也不直接减少磁盘码流，应与部署剪枝分开。

“可证明剪枝”还需读清楚被保证的对象。Coreset 预印本 [38] 明确指出，无约束查询族下不存在一般的非平凡乘法近似保证；其正向结果依赖指定分辨率/查询集合、固定目标贡献，以及有效性和对数透射率稳定等条件。从有限查询目标到真实重新合成图像的保证，需要额外假设。不能把线性代理、固定视图上的保证推广为任意新视角、任意遮挡条件下的全局保证。

## 8.5 从全局排序到带保护的渐进剪枝

全局 top-$K$ 简单，但预算常集中于面积大或纹理丰富的区域。空间块、深度层、anchor 或语义对象的局部预算可保护小目标，却可能给空旷区域浪费预算。结构化剪枝删除整个 anchor、SH 阶或 LOD，较易减少实际访存与分支；非结构化删除更灵活，但若只把门设为零、不压实数组，渲染器仍可能处理原来的索引。

稳健流程以不参与恢复梯度的验证视角设置接受条件，并保留独立最终测试。校准相机应覆盖预期路线、遮挡解除视角和不同尺度；无法观测的背面不应被默认成“无价值”。对环绕观看，若训练数据本身缺少背面，既不能靠剪枝分数补出真值，也不能声称环绕质量已经得到保证。

**Listing 8.1** 含风险复核与回滚的渐进压实。

```
scene = load_frozen_scene()
for target_count in schedule:
    snapshot = save_scene_and_optimizer(scene)
    scores = measure(scene, calibration_views)
    keep = select(scores, target_count, spatial_detail_guards)
    scene = compact_attributes_and_state(scene, keep)
    rebuild_anchor_ids_and_tile_lists(scene)
    recover(scene, training_views, fixed_step_budget)
    report = evaluate(scene, validation_views)
    if violates_quality_or_detail_budget(report):
        scene = restore(snapshot)
        break
save_codec(scene)
evaluate(decode_codec(), final_test_views)
```

恢复阶段需同步裁切梯度、优化器矩和每个高斯的历史统计；否则旧索引可能把状态写回错误高斯。删除 anchor 后需处理空子集、父子映射和 LOD 索引。每轮恢复会改变尺度、位置与透射率，因此下一轮应重新统计重要性；复用最初的固定排序不再反映当前模型。

保护措施可包括每块最低覆盖、细结构 mask、深度边界误差和逐视角最大劣化阈值。保护本身也可能过度保守，应给出消融：无保护、仅空间保护、加边界保护分别消耗多少高斯，改善哪些失败案例。保护规则由验证集设计后，最后在独立测试集报告；不能把不断查看测试误差的过程称为盲测。

## 8.6 删除之后还要重查排序与系统收益

仅删除且位置不变时，原有正确排序的相对顺序仍正确；但缓存的 tile 列表、计数、指针和高斯编号通常需要重建。恢复微调或后续位置量化一旦改变深度，必须按新属性重新投影和排序。对深度相近的透明层，微小排序交换也可能产生可见差异。检查平移、环绕和前后景交界处的闪烁，比只检查一帧平均分数更有效。

逐项验证还包括：空块和尾块可否解码，早停是否过早饱和，细杆是否断裂，背面揭露后是否出现洞，颜色是否泄漏到背景，深度和法向是否退化。少量高斯被替换为巨大 footprint 时，$N$ 虽下降，tile 重复引用与像素过绘制却可能上升。可用如下近似理解工作量：

$$C_{\text{frame}} \approx c_p N_{\text{visible}} + c_s M_{\text{tile}} \log M_{\text{tile}} + c_a A_{\text{blend}}, \tag{8.9}$$

$M_{\text{tile}}$ 是高斯 –tile 引用数，$A_{\text{blend}}$ 是实际合成访问量；眞实排序 kernel 可能采用 radix sort，复杂度与键位数有关，这里的对数项只作一般比较排序示意。实际结论需由 GPU profiler 和端到端帧时间确认，不能只从 $N$ 推断加速倍数。

剪枝与量化应联合验证而非假设收益相加。删除部分重叠高斯后，剩余位置、SH 与不透明度可能更敏感；量化范围也随子集改变。反过来，提高保留高斯的位宽可能在相同文件预算下优于保留更多低精度高斯。先固定一项扫描另一项，再比较相同完整字节预算的工作点，才能看清这种交换。

**本章回顾**

删除风险来自透射率、前后景颜色与多基元交互。贡献分数、梯度和重叠都能帮助筛选，但不能替代删除后眞实合成。小步压实、持出保护、恢复预算、索引重建和独立测试把启发式剪枝变为可审计的工程流程。

**练习与提示**

1. 按式 (8.4) 构造两个 opacity 相同、删除误差相差很大的基元。提示：改变后景颜色而非只改变前景颜色。
2. 十层不透明度均为 0.1 时，计算删一层与删五层的总不透明度变化。为什么不能把每层变化线性相加？
3. 用两个完全重叠基元说明：三维 Gaussian Gram 矩阵秩低不等于 alpha 合成可无损剪枝。尝试在三个不同 $G(u)$ 值上求一个统一的替代不透明度。
4. 设计一组消融，分离剪枝分数、恢复训练和 SH 蒸馏的作用；写明哪组视角承担调参、哪组承担最后测试。
5. 若删除一半高斯后 FPS 只上升 5%，你会检查哪三个 kernel 指标？若高斯数减少但 tile 引用增加，可能发生了什么？

# 第 9 章　学习式压缩、熵模型与渐进传输

上一章通过删除基元减少表示规模，第 7 章通过量化减少单个属性的精度。本章继续处理量化后的符号：如果相邻高斯或同一 anchor 的属性存在相关性，怎样用更少的位描述它们？读者需要同时跟住两条链：一条是概率模型怎样降低交叉熵，另一条是解码器怎样仅凭已收到的信息还原符号。只有两条链都成立，学习到的码率收益才能成为可用文件。

## 9.1　量化与熵编码分别改变什么

学习式 codec 可概括为

$$\mathcal{G} \xrightarrow{\text{结构化/变换}} y \xrightarrow{Q} k_{1:M} \xrightarrow{\text{概率模型与无损编码}} b. \tag{9.1}$$

结构化可以建立 anchor、空间块与局部特征；量化将连续变量变成离散符号；熵编码再利用符号的非均匀分布和上下文相关性。解码必须先还原同一串整数，再反量化并恢复可渲染表示。对于固定量化结果，熵编码是无损阶段，不应进一步改变渲染图像。编码器若还联合改变变换、属性或量化步长，整个 codec 仍然是有损的。

必须区分两类网络。**表示解码器**由 anchor 特征生成尺度、旋转、颜色等渲染属性；**熵模型**为待解码符号生成概率。二者都可能使用小型 MLP，但功能不同。一个只在编码/解码时使用的概率网络，不必在每帧渲染时运行；一个视角条件的高斯生成器则可能每帧执行。报告解码开销和 FPS 时，应指出网络运行在哪个阶段。

最简单的固定长索引为每个符号分配 $\lceil \log_2 K \rceil$ bit。若 $K = 4$ 且四个值的概率为 $(1/2, 1/4, 1/8, 1/8)$，更常见的符号理应获得更短描述。算术编码或范围编码将一整串符号映射成一个区间中的数，避免逐符号整数码长的限制；它们需要的是离散概率质量，不是连续概率密度。

## 9.2　从连续分布得到符号概率

设变换域标量 $y$ 使用步长 $\Delta > 0$、实数偏移 $a$，编码与恢复为

$$k = \operatorname{round} \frac{y - a}{\Delta}, \qquad \hat{y} = a + \Delta k.$$

假设熵模型依据可用上下文 $c$ 输出连续分布的累积分布函数 $F_\psi(\cdot \mid c)$，则未裁剪的第 $k$ 个格子有概率

$$\begin{aligned} p_\psi(k \mid c) &= \int_{a+\Delta(k-1/2)}^{a+\Delta(k+1/2)} f_\psi(t \mid c)\, dt \\ &= F_\psi\big(a + \Delta(k + 1/2) \mid c\big) - F_\psi\big(a + \Delta(k - 1/2) \mid c\big). \end{aligned} \tag{9.2}$$

高斯模型的条件均值、标准差可由网络预测；此处描述的是属性数值的统计分布，不要把它们与三维基元的空间中心和协方差混为一谈。混合高斯则对各分量的 CDF 差按混合权重求和。

例如标准正态分布、$a = 0, \Delta = 1$ 时，零符号概率为 $\Phi(0.5) - \Phi(-0.5) \approx 0.382925$，而零点密度约为 0.398942。密度可大于一、也带有坐标单位，不能直接当符号概率送入编码器。步长变小时，同一点的格子概率通常下降、符号数增加，码率与失真由此产生交换。

有限符号范围还需要处理尾部。若量化器把范围外数值裁剪到两端，端点符号的概率应包含被裁剪的整个尾部；若协议采用逃逸符号，则须编码逃逸事件及其残差。只对中间格子的概率做截取

而不处理剩余质量，会使概率和不等于一。极小标准差、浮点下溢和数值相近的 CDF 相减也可能产生零概率；应设置合法尺度范围、稳定计算方式与明确的尾部协议。

训练中常用均匀噪声替代舍入，以得到可微的率失真代理。它让网络适应量化扰动，但部署时仍应使用真实整数和式 (9.2) 对应的离散概率。噪声代理损失下降，不足以证明实际量化失真或码流长度下降。

## 9.3 上下文必须先于当前符号到达

对符号序列 $k_{1:M}$，一种因果分解是

$$\begin{aligned} p_\psi(k_{1:M} \mid h) &= \prod_{m=1}^{M} p_\psi(k_m \mid k_{<m}, h), \\ R_{\text{est}} &= -\sum_{m=1}^{M} \log_2 p_\psi(k_m \mid k_{<m}, h). \end{aligned} \tag{9.3}$$

$h$ 是解码器已拥有的共享或侧信息。它可以来自先解码的位置、hash grid、超先验或前一层结果，但不能来自编码器私有的未量化属性。若侧信息本身需要传输，它的字节也进入总账本。空间上“离得近”不意味着解码时“已经可用”。

一个合法的通用依赖顺序是：先读取头部与模型标识，再解码空间结构和必要侧信息，随后解码特征组 1，利用组 1 解码特征组 2，最后恢复属性并渲染。实际写文件时，编码器必须用已经量化/重建的组 1 预测组 2；若偷偷使用原始浮点组 1，解码器会得到不同概率，算术解码可能从第一个不一致符号起全部失步。训练也应模拟解码可用的上下文；若采用噪声代理，最后须检查它与真实舍入上下文的差异。

**Listing 9.1** 因果解码的基本约束。概率由已还原状态产生。

```
header, side_info = read_header_and_side_streams(file)
model = load_verified_model(header.model_id)
state = init_state(side_info)
for block in header.block_order:
    reset_coder_if_required(block)
    for group in dependency_order(block):
        params = model.predict(state, group)
        cdf = make_integer_cdf(params, header.cdf_rule)
        symbols = entropy_decode(file, cdf, group.count)
        values = dequantize(symbols, group.quantizer)
        state.update(group, values)
scene = reconstruct_valid_attributes(state)
```

逐符号自回归能利用较多上下文，却可能限制并行解码。分组、通道分块或棋盘式依赖设计，可让同组符号在给定先验下并行处理；独立空间块还可支持随机访问。代价可能是上下文变少、侧信息重复和重置开销。因此“更强概率模型”应由实际码流收益与解码吞吐共同判断，不能仅比较训练交叉熵。

## 9.4 交叉熵、有限精度 CDF 与文件长度

若真实离散分布为 $P$、采用模型为 $Q$，则期望码率代理满足

$$\mathbb{E}_P[-\log_2 Q(k)] = H(P) + D_{\text{KL}}(P \| Q). \tag{9.4}$$

这说明拟合概率可以减少模型失配，但式 (9.3) 是对指定符号序列的理想位数，不是文件系统返回的字节数。实际编码器通常使用有限精度整数 CDF。记精度为 $P_{\mathrm{cdf}}$、总频数为 $T = 2^{P_{\mathrm{cdf}}}$，则

$$\begin{aligned} 0 = C_0 \le C_1 \le \cdots \le C_K = T, \qquad T = 2^{P_{\mathrm{cdf}}}, \\ \widetilde{p}_k = \frac{C_{k+1} - C_k}{T}. \end{aligned} \tag{9.5}$$

会出现的符号必须有正整数频数；零频数符号无法编码。将连续概率分配到整数频数时，要约定最小频数、归一化余数的分配、逃逸范围与并列处理。提高 $P_{\mathrm{cdf}}$ 能减小部分概率近似误差，却增加表和运算成本，并受编码器整数范围约束。

把 $p_\psi$ 换成 $\widetilde{p}$ 后，有限精度概率的交叉熵一般会改变；编码器还会产生终止位、块边界与字节填充。因此可把完整成本分为

$$\begin{aligned} B_{\mathrm{file}} = B_{\mathrm{header}} + B_{\mathrm{structure}} + B_{\mathrm{model/codebook}} + B_{\mathrm{side}} \\ + B_{\mathrm{entropy\ payload}} + B_{\mathrm{directory/checksum}}. \end{aligned} \tag{9.6}$$

其中负载项是实际写出的字节，已经包含该实现的终止与填充，不能再重复相加。若希望诊断开销，可额外列出负载内部的理想交叉熵、有限精度交叉熵和实现差额。差额不是对所有编码器、块长和数据都固定的一个常数。

浮点网络在不同设备上产生的微小差异，也可能改变整数 CDF 的取整结果。协议应使用可复现的概率计算与 CDF 规则，或保存必要的表、限定实现与精度并执行跨设备往返测试；“使用同一权重”本身不能证明熵编码兼容。损坏或不匹配的流应被拒绝，不能静默输出一组数值合法但内容错误的高斯。

### 9.4.1 一个只有 22 byte 的教学文件

考虑序列 $(0, 0, 0, 0, 1, 1, 2, 3)$，概率为 $(1/2, 1/4, 1/8, 1/8)$。采用前缀码 $0 \mapsto$ `0`、$1 \mapsto$ `10`、$2 \mapsto$ `110`、$3 \mapsto$ `111`，得到

$$\texttt{00001010110111} \qquad (14\ \text{bit}).$$

理想负对数概率和也是 $4 \times 1 + 2 \times 2 + 1 \times 3 + 1 \times 3 = 14$ bit。按从高位到低位打包、末尾补两个零，负载为十六进制 `0adc`，恰为 2 byte。若教学协议再保存 16 byte 头部和 4 byte 码长表，完整文件就是 22 byte，而非 14/8 byte。

这里的 16 byte 可具体定义为：4 byte 魔数、1 byte 版本、1 byte 编码器编号、2 byte 标志、4 byte 符号数和 4 byte 有效负载位数；码长表为 `01020303`，字母表顺序固定。码长表按已约定的规范前缀码规则恢复码字，符号数和有效位数使解码器能区分数据与填充。这个微型协议没有校验和或随机访问，不是生产格式，也不是本次真实场景的熵压缩实验。

**Listing 9.2** 用断言核对教学负载的 14 bit、2 byte 及十六进制值 0adc。

```
codes = {0: "0", 1: "10", 2: "110", 3: "111"}
symbols = [0, 0, 0, 0, 1, 1, 2, 3]
bits = "".join(codes[k] for k in symbols)
padding = (-len(bits)) % 8
payload = int(bits + "0" * padding, 2).to_bytes(
    (len(bits) + padding) // 8, "big")
assert len(bits) == 14 and payload == bytes.fromhex("0adc")
assert 16 + 4 + len(payload) == 22
```

再看较大教学预算：$10^6$ 个符号的平均交叉熵为 3.7 bit/symbol，理想负载为 462 500 byte。若码本为 2 MB，总和已有 2 462 500 byte；若指 2 MiB，则是 2 559 652 byte。这两个数都还不是完整文件大小：模型、侧信息、有限精度编码差额和容器开销尚未给出。对很短的块，模型与头部甚至可能比符号负载更大。

## 9.5 Anchor、hash context 与混合熵模型

### 9.5.1 HAC 与 HAC++：概率上下文不等于替代表示

Scaffold-GS 用 anchor 及其特征、尺度和局部偏移生成附近高斯 [21]。HAC 在此类表示上引入二值多分辨率 hash grid：由 anchor 位置查询 hash 特征，用上下文网络预测量化步长和属性分布的均值、尺度，再编码量化后的 anchor 属性 [39]。hash 特征用于预测概率，并非直接取代所有 anchor 特征。hash grid 本身、anchor 位置、mask 和网络也要保存或通过明确共享资源提供。

HAC 的自适应步长把不同 anchor/属性的失真与码率联系起来；mask 则去掉无效局部高斯。HAC++ 在 hash context 之外增加 anchor 内部的特征分块上下文，并改进自适应 masking[40]。已解码通道块帮助预测后续块，使它与式 (9.3) 的因果关系一致。不能把两者概括成“hash 压缩了所有属性”而省略条件分布、侧信息与解码顺序。

其收益包含表示、量化、mask 与熵模型的共同作用。若要测量上下文本身的贡献，应固定量化符号与 mask，比较独立模型、hash context 和增加内部上下文后的实际位数；若训练过程也改变了场景，则另报告完整率失真结果。本文依据本地 HAC v3、HAC++ v4 论文及官方元数据整理机制。

### 9.5.2 HEMGS：先验与自回归分别解释相关性

HEMGS 的混合模型包括可变率预测器、超先验网络与自回归网络 [41]。可变率预测器用率参数控制量化步长；先验结合冻结的跨场景三维特征提取器与场景专属网络；自回归部分从已编码的邻域元素构造上下文，并依据局部稀疏程度选择感受野内的可用邻居。其压缩顺序先提供量化位置，再压缩局部特征，最后利用已还原特征帮助编码尺度与偏移。

“混合”在这里有明确的结构含义，不是随意为几种属性各选一个分布。论文中的冻结模型可以避免每场景重复训练/存储相同权重，但部署仍需要获得它；场景专属部分也仍有字节成本。可变率模型能在一个训练框架中提供多个率失真工作点，不表示任意截断码流都可解码，更不表示无需场景优化。本节按 HEMGS v2 说明机制。

## 9.6 前馈 codec、图像条件解码与跨场景生成

“前馈”只说明一次计算如何执行，不能单独说明输入与任务。表 9.1 按已核验论文版本区分三条容易混淆的路线。

**表 9.1** 相关系统的输入与输出。此表比较任务定义，不排列跨论文性能。

| 方法及所据版本 | 输入与核心机制 | 解释边界 |
|---|---|---|
| FCGS，v3 | 已有 3DGS；多路径熵模块、基元间与基元内上下文，前馈生成压缩表示 | 测试时压缩无需每场景优化；场景原始重建仍有成本 |
| ICGS-Quantizer，v2 | 网格化场景块；几何/纹理潜变量与共享残差码本，解码可选当前图像条件 | 共享模型与码本需可用；图像条件更新不等同于只凭图像生成全部场景 |
| GSCodec Studio，v1 | 模块化 3D/4D 表示、压缩与渲染组件，含静态及动态组合 codec | 是实验与系统框架，具体压缩、训练和输入条件由组件组合决定 |

Fast Feedforward 3D Gaussian Splatting Compression 提出的 FCGS，以已重建的高斯集合为输入，通过学习到的变换与多路径熵模型快速压缩 [42]。其基元间上下文使用先解码批次，基元内上下文使用先解码通道块。它可以研究跨场景 codec 泛化，但不是把少量输入图像直接重建成新场景的通用生成器。比较编码时间时，应明确是否包含原始 3DGS 训练、预训练 codec 和数据搬运。

Image-Conditioned 3D Gaussian Splat Quantization 则先把场景置于稀疏网格，用局部块联合编码高斯及属性，再以共享码本对几何与纹理潜变量做残差向量量化 [43]。其解码器可以引入当前图像的多尺度 DINOv2 特征，通过可见性感知的特征融合适应照明或外观变化；无需更新时也可不带图像解码归档场景。这里的图像是有既存场景码的条件信息。若评测使用这些新图像，必须说明其数量、相机/对齐条件、字节与特征提取成本，以及哪些图像仍作为独立测试。

GSCodec Studio 汇集静态/动态表示、训练中压缩模拟、后训练压缩与渲染组件，并给出组合 codec[44]。它适合在统一实现中做组件消融，但框架名称本身不能作为“跨场景图像前馈生成”的机制证据。

对以上路线，分别记录模型在何种场景上训练、测试时是否迭代、需要哪些输入、共享资源如何计费。跨场景泛化是对未参与训练场景的实证结论；它与 codec 是否前馈、是否可变率、是否渐进是四个不同维度。

## 9.7 渐进码流：可复用的前缀与可渲染的层

多个独立率失真模型各输出一个完整文件，并不构成渐进传输。设文件由头部 $H$、基础层 $b_0$ 与增量层 $b_1, \dots, b_L$ 组成，可定义可用前缀

$$\begin{aligned} b^{(\ell)} &= H \| b_0 \| \cdots \| b_\ell, \qquad b^{(\ell)} \preceq b^{(\ell+1)}, \\ R_\ell &= 8|b^{(\ell)}|_{\text{byte}}, \qquad R_0 \le R_1 \le \cdots \le R_L. \end{aligned} \tag{9.7}$$

$\preceq$ 表示字节串前缀关系；若层分文件，则可对已收到的层集合写包含关系。累计位数 $R_\ell$ 是数值，只能比较大小，不能写成 $R_1 \subset R_2$。还应区分累计 $R_\ell$ 与增量 $r_\ell = 8|b_\ell|$；评估高质量层时不能只报最后一层的增量。

每个声明为可播放的前缀，必须具有完整依赖和合法的表示：基础位置与模型是否已到达？四元数是否可归一化？缺失的 SH/残差如何置默认值？码字是否在层边界正确终止？随机截去文件最

后若干字节通常不满足这些要求。多层训练可优化

$$\mathcal{L} = \sum_{\ell=0}^{L} w_\ell \big[ D_\ell + \lambda_\ell R_\ell \big], \qquad w_\ell \geq 0,$$

使早期层也承担渲染质量目标。即使保留集合随层增加，alpha 遮挡和量化精化仍可能使某些视角变差，不能从前缀关系推出 PSNR 严格单调。

PCGS 在 anchor 表示上联合处理数量与质量 [45]。数量侧使用随层单调的 mask，使已出现的 anchor/高斯在后层继续存在；质量侧先对新出现属性做初始量化，再对已有量化格进行三等分精化。旧恢复值和旧步长给出合法区间，新层只需指出三个子区间之一，并利用跨层上下文预测概率。这里的 trit 是三元细化选择，不是“再发送一个完整高精度浮点数”。本文按 PCGS v1 整理。

数量渐进、属性精化与视角优先传输可以组合，但它们解决不同问题。PCGS 的层级可复用不自动提供任意空间块的随机访问；按视锥请求也需要处理遮挡解除、相机转向和跨块依赖。评价时除累计字节与每层质量，还应记录基础层到首帧、每次增量的解码时延、切层闪烁和中断后可继续使用的最后完整层。

## 9.8 把概率模型写成版本化协议

可维护 codec 至少需要明确下面六组信息。它们可合并存储，但不能只留在编码脚本的默认参数里。

1. **身份与语义：**魔数、版本、编码器/模型标识与校验，世界坐标归一化、单位、属性顺序、SH 基及方向、颜色空间和背景约定。
2. **结构与量化：**高斯或 anchor 数、空间块与父子关系、mask、量化域、偏移/零点、步长、码本、位序、字节序及异常值协议。
3. **概率与依赖：**CDF 精度及构造规则、符号支持集、上下文顺序、分组边界、共享网络版本、必须先解码的侧信息。
4. **层与目录：**基础层/增量层标识、每层有效长度、块偏移与包围盒、依赖列表、重置点、可用前缀和随机访问条件。
5. **完整性与边界：**校验和、总数和长度上限、截断检测、未知版本处理、空块、尾块、损坏概率表与非法属性的拒绝规则。
6. **恢复接口：**反量化、旋转/尺度合法化、属性展开或即时查表、GPU 上传、索引重建和 viewer 所需的渲染语义。

真正的随机访问要求目标块的依赖闭包可获得。独立块可重置熵编码状态，并保留局部概率侧信息；若仍依赖此前全部块的自回归状态，就只能顺序解码，除非保存检查点。减小块通常降低首次请求成本，却增加头部、重复模型与重置损失，应同时画请求字节、质量和时延曲线。

实验应在新进程中只读取发布文件与声明共享资源，验证符号逐项一致、属性合法且图像一致。再测模型加载、熵解码、属性重建和上传等阶段，最后测实际首帧与稳定渲染。对现成 viewer，通常需要展开成其支持的属性语义；保留 anchor 生成器或压缩域查表渲染时，应记录额外 kernel、常驻模型和缓存。第 12 章区分本次已执行的合成量化/剪枝 smoke test 与尚未运行的学习式熵 codec 复现，不能把两者混写成同一验证。

**本章回顾**

熵模型为量化格分配离散概率，上下文只能来自已可用信息，实际编码还需要有限精度 CDF 与确定的解码协议。HAC、HEMGS、FCGS、图像条件量化和模块化框架的输入、共享资源与训练条件各不相同。渐进传输要求可渲染前缀与依赖闭合；完整字节、首帧时延和随机访问成本共同决定部署价值。

**练习与提示**

1. 用标准正态 CDF 计算 $\Delta = 1$ 时 $k = 0, 1, -1$ 的概率；若只保留这三个符号并把尾部裁剪到端点，端点概率应怎样改变？提示：端点格须包含无穷尾部。
2. 若一个网络用未量化的位置预测属性概率，编码端与解码端会遇到什么问题？画出修正后的依赖顺序，并指出需要先发送哪些量。
3. 对 22 byte 教学文件，把八个符号全部改为零。比较原概率模型与重新拟合模型的负载收益，解释为何短序列上更新模型未必节省文件总字节。
4. $10^6$ 个符号平均交叉熵为 3.7 bit/symbol、码本为 2 MiB，为什么不能直接宣布文件为 2 559 652 byte？至少补出四类尚缺成本，并说明哪些应通过实际文件测量。
5. 设计两层场景传输：基础层提供可用轮廓，增量层改进高光。写出每层的数据与依赖，分别报告增量和累计字节；不要预先承诺某个 FPS，先列出实现和测量条件。
6. 将自回归场景划成独立空间块，怎样实现相机转向后的随机访问？提示：同时考虑概率状态重置、全局侧信息、块目录和遮挡解除后的质量。

# 第三部分

# 实验协议与研究实践

# 第 10 章　可复现训练、部署评测与排错

一个方法的实现说明了如何生成结果；评测协议决定结果能支持什么结论。本章以同一个场景、同一组相机和同一个 renderer 为起点，区分表示误差、数据处理差异、恢复训练收益与系统实现收益。协议应在选择位宽和剪枝阈值前写下，而不是看见最好结果以后补写。

## 10.1　把图像、相机与尺度一起固定

保存原始图像及校验值、去畸变过程、内参、世界到相机的外参、训练/校准/测试视角清单，并记录分辨率、裁剪、像素中心约定和颜色空间。图像先裁剪再缩放时，主点先减去裁剪偏移，再随缩放系数变化；只缩放焦距而保留原主点，会把几何错误交给模型补偿。非针孔相机应在对应模型中生成射线，或先去畸变后更新内参，不能只把畸变系数丢掉。

对场景归一化，显式记录

$$\mathbf{x}' = s_0 R_0(\mathbf{x} - \mathbf{c}_0), \qquad s_0 > 0. \tag{10.1}$$

高斯均值、相机中心和协方差必须随之变换：$\Sigma' = s_0^2 R_0 \Sigma R_0^\top$。密度若原先按世界长度积分，则在新长度单位下变为 $\sigma' = \sigma / s_0$，使光学厚度 $\sigma\,\delta$ 保持不变。高斯的无量纲 opacity 不按这个规则缩放。在缩放前后渲染相同物理射线，是检查坐标变换的一种有效方式。

COLMAP 的重建、相机和图像 ID 应保持可追踪。把稀疏点重投影回图像后，检查正深度、像素误差和轮廓位置；绘出相机中心与光轴，能比损失曲线更快地发现外参方向翻转。NeRF-W 处理非受控图像集合的曝光与瞬态变化，但引入外观建模并不等于相机错误已被解决 [15]。

**适用条件与误区**

测试相机参与剪枝评分、码本训练或位宽选择后，就已经成为校准数据。沿连续视频序列随机抽相邻帧，可能得到非常容易的“新视角”；应记录帧间距、相机覆盖和留出规则。不同方法使用不同训练图像、背景或输入分辨率时，先复现统一协议，再讨论表示本身的差异。

## 10.2　从训练前的检查到冻结 checkpoint

先用少量射线或低分辨率图像验证损失下降、背景正确、梯度有限，再增加训练预算。对 3DGS，额外记录初始化点数、球谐最高阶、位置与尺度学习率、densification 起止步、clone/split/prune 阈值和 opacity reset。这些设置改变高斯数和可压缩性，不能被当成与 codec 无关的细节。

压缩实验先保存未压缩 checkpoint 的哈希值，再固定评测相机。PTQ 和有恢复训练的方法各设一组：后一组报告恢复步数、时间和额外训练数据。若方法使用蒸馏，分别以原模型与真实观测图像为参照，避免把“接近老师”写成“几何更准确”。Nerfstudio 的模块化设计有助于拆开数据处理、表示和采样组件，但工具包中的默认配置仍需要锁定版本 [46]。

## 10.3　图像指标的计算对象与平均方式

对动态范围为 $[0,1]$ 的三个颜色通道，设评测像素集合为 $\mathcal{M}$，定义

$$\mathrm{MSE} = \frac{1}{3|\mathcal{M}|} \sum_{p \in \mathcal{M}} \left\| \hat{\mathbf{c}}_p - \mathbf{c}_p \right\|_2^2, \qquad \mathrm{PSNR} = -10 \log_{10}(\mathrm{MSE}). \tag{10.2}$$

零误差时 PSNR 为正无穷，不要随意加一个很大的有限值参与平均。逐图 PSNR 的算术均值，与先汇总所有像素的 MSE 再取对数，一般不同；图像尺寸不同时，后者还按像素数加权。报告必须注明平均单位。裁掉边界、忽略背景或设置前景掩码都会改变指标，掩码应对所有方法一致。

SSIM[47] 比较局部亮度、对比度与结构；需要记录窗口大小、核、动态范围和是否逐通道平均。LPIPS[48] 是带特定网络权重的学习感知距离，报告网络版本、权重、输入归一化和裁剪规则。两者都不能由 PSNR 换算，也不能在没有相应模型的环境中填入“估计值”。相机曝光未对齐时，任意拟合每张测试图像的颜色变换会掩盖外观误差；若采用对齐，必须将未经对齐与对齐后的结果分开，并说明拟合的数据来源。

## 10.4 几何、时间一致性与任务指标

外观相似不保证高斯中心位于真实表面。几何评测可使用有真值的深度误差、法向角误差、点到面距离或明确阈值下的重建精确率与召回率。深度要说明是相机 $Z$、沿射线距离、未归一化期望，还是条件于非背景命中的深度。无效像素、缺测区域和坐标配准规则不能在不同方法间改变。如果允许用真值拟合每个场景的尺度，应把它作为额外条件报告。

动态场景先固定时间戳和相机轨迹，再计算时序差异。连续两帧的 RGB 差不自动等于闪烁，因为真实运动和遮挡也会产生变化；可使用可靠对应区域的时间一致性误差，并把无对应区域单独可视化。机器人或 AR 任务还应指定跟踪、轮廓、碰撞或定位指标。RGB、几何与任务指标构成一组约束，不能未经验证就相互替代。

## 10.5 存储、内存与时延的测量边界

**表 10.1** 每个系统数字都需要明确计费边界。

| 报告对象 | 需要包含与说明的内容 |
|---|---|
| 磁盘码流字节 | 实际文件长度；码本、索引、预测器、头部、分块表与必要侧信息 |
| 编码耗时 | 从冻结场景到完整码流；分列范围/重要性统计、码本拟合、可选恢复与位流生成，注明文件写入边界 |
| 解码耗时 | 从完整码流到可渲染属性；区分内存解码、文件读取加解码和 GPU 上传，不能只计反量化算子 |
| 解码后 CPU 内存 | 模型属性、码本、索引；是否包含框架、文件缓存和解码工作区 |
| 渲染显存峰值 | 属性、tile 副本、排序键和工作区；说明 allocated 与 reserved |
| 冷启动端到端时延 | 读文件、解码、上传、首次分配与首帧；网络下载另计或明确包含 |
| 稳态每帧时延 | 固定分辨率和轨迹下的投影、分桶、排序、合成；说明是否包含调度和显示 |

读取小文件并不保证低运行内存：一个紧凑 codec 可能完整展开为 float32 属性。本书的压缩比定义为 $B_{\text{baseline}}/B_{\text{total}}$：分子是锁定格式的基线字节，分母是压缩后包含全部必要侧信息的完整字节。“相对于 FP32 属性数组”与“相对于含文本头部的 PLY”是两种不同基线，不能在同一比较中替换；也不能只用属性净荷替代分母。共享模型若预装在设备上，可以另报告摊销情形，但必须同时给出模型完整大小、场景数及版本要求。

编码计时至少区分每场景准备与位流生成。前者包括量化范围统计、重要性评分、码本学习，以及方法所需的剪枝恢复或 QAT；后者包括量化、索引打包、熵编码和侧信息序列化。单报位流生成时间时，仍应另列准备成本和每场景总耗时。跨场景共享 codec 的预训练与原始重建训练另报，不

得把它们混入某方法的计时、又从另一方法中省去。文件读取和写入、CPU/GPU 搬运是否包含，也要逐项声明。如果解码输入已在内存中，应称为内存解码；操作系统缓存命中的文件读取加解码，也不等于冷盘加载。第 12 章的教学实验按这些边界分别保留计时样本。

GPU 计时需预热并同步。CUDA event 适合记录设备区间，单调时钟适合记录完整的 CPU 发起到 GPU 完成。计时区间前的同步排除先前工作，区间后的同步等待当前工作完成；不要把异步 kernel 发射时间当作渲染时间。报告重复次数、预热次数、中位数和尾延迟，并注明功耗模式、GPU、驱动、软件提交、分辨率和相机轨迹。吞吐量 FPS 应由完整测量区间的帧数除以时间得到；它不是有排队、加载或并行解码时的用户感知延迟倒数。

**Listing 10.1** 同步区间的示意。生产测量还需记录配置和原始逐帧时间。

```
for _ in range(warmup):
    render(scene, camera)
torch.cuda.synchronize()
t0 = time.perf_counter_ns()
for camera in fixed_cameras:
    render(scene, camera)
torch.cuda.synchronize()
elapsed_s = (time.perf_counter_ns() - t0) / 1e9
fps = len(fixed_cameras) / elapsed_s
```

## 10.6 用匹配预算的消融回答因果问题

假设系统包含剪枝 $P$、量化 $Q$、恢复训练 $F$ 与熵编码 $E$。基线、$P$、$Q$、$P+Q$ 可检查主要交互；比较 $P+Q$ 与 $P+Q+F$ 才能估计恢复收益。同一解码表示分别有无 $E$，可检查熵编码是否只改变码率与解码成本。单一路径 $1 \to P \to P+Q \to P+Q+F \to P+Q+F+E$ 不足以为所有组件独立归因。

预算也有多个维度。固定总字节来比较质量，固定质量阈值来比较总成本，或在固定高斯数下比较属性编码；不要把“更多恢复步数”“更高分辨率”悄悄混入一行。场景数少时展示每个场景的结果与失败案例，比只给一个均值更有解释力。多种子训练按同一场景配对报告变化，测试视角数量不能冒充独立场景样本数。

## 10.7 排错从可证伪的假设开始

**表 10.2** 观察、假设和独立检查。

| 观察 | 候选解释 | 下一项检查 |
|---|---|---|
| 重建漂浮、深度反号 | 外参翻转、单位不一致 | 重投影已知点；可视化相机光轴；关闭可学习外观 |
| 远景或细边缘闪烁 | 点采样混叠、投影近似、排序切换 | 固定相机尺度扫频；比较过滤配置；绘误差随视角曲线 |
| 文件变小而 FPS 不变 | 全量解码后 primitive 数和覆盖不变 | 比较展开字节、tile 副本数、排序区间和合成区间 |
| 删点后渲染更慢 | 剩余大 footprint、tile 负载失衡 | 画 footprint 分布、每 tile 候选数；检查恢复训练是否放大尺度 |

| 观察 | 候选解释 | 下一项检查 |
| --- | --- | --- |
| 量化后背景穿透 | opacity 端点、尺度变小或近似排序变化 | 单属性替换；保留前景结构；逐像素检查透射率 |
| 首次加载慢 | 串行上下文、工作区分配或上传 | 分离文件读、解码、分配、上传和首帧计时 |

可定义“影响可见像素的候选对占比”来诊断无效工作，但须固定显著性阈值。压缩并不必然提高该比值；只节省磁盘字节的方法也可能完全不改变候选对。先测量瓶颈所在，再改变表示或实现。

**练习与提示**

1. 两张相同尺寸图像的 MSE 分别为 $10^{-2}$ 和 $10^{-4}$。逐图平均 PSNR 与总体 MSE 的 PSNR 各是多少？提示：先平均再取对数与反过来不是同一操作。
2. 某 codec 文件缩小 20 倍、解码后属性不变、FPS 不变。这个结果能支持什么结论？提示：分别讨论传输、冷启动与稳态渲染。
3. 为 $P, Q, F, E$ 设计四项配对消融，各注明控制变量与一个可推翻结论的观察。

# 第 11 章　从已核验工作提出研究问题

研究地图的作用是指出可检验的缺口。把论文名称按年份排成长表，并不能说明它们解决了同一个问题。本章只围绕表示、误差与系统接口讨论有证据的方法，本次来源核查日期为 2026-10-21 至 22；未独立复现的论文结果均不写成本书实测。与一张跨数据、跨硬件的排行榜相比，明确输入条件与失败标准更能帮助选择实验。

## 11.1　把论文贡献放回统一接口

给定输入图像与相机，训练过程产生场景参数 $\Theta$；编码器产生字节流 $b$；解码器还原可渲染表示 $\hat{\Theta}$，最后在相机 $v$ 上产生图像。若方法还需要侧图像、共享网络或外部码本，应把它们作为显式输入：

$$b = E(\Theta; \psi), \qquad \hat{\Theta} = D(b, \psi, s), \qquad \hat{I}_v = \mathcal{R}(\hat{\Theta}, v). \tag{11.1}$$

这里 $\psi$ 是共享模型，$s$ 是场景侧信息。只有接收端确实拥有同一版本的 $\psi$ 和 $s$，才可单独传 $b$；共享模型的预装或摊销条件必须报告。跨场景前馈压缩不意味着从少量图像直接生成新场景，二者输入不同。

**表 11.1**　方法应比较其作用接口，而非把标题里的倍率直接排序。

| 路线 | 减少或重分配什么 | 必须额外检查什么 |
|---|---|---|
| LightGaussian | 重要性剪枝、SH 处理与属性编码 | 剪枝后的恢复训练、视角覆盖和实际码本成本 |
| Scaffold-GS | 用 anchor 及条件预测组织基元 | 特征/网络大小、视角条件计算与随机访问 |
| HAC / HAC++ | 用结构化上下文降低属性的条件熵 | hash/网络等侧信息、解码顺序及预算分母 |
| Fast Feedforward 压缩 | 将已训练场景快速变成紧凑码流 | 共享训练数据与模型、未见场景的分布差异 |
| PCGS | 渐进式传输与表示层次 | 首帧质量、逐层累计字节、随机访问和层间依赖 |
| GSCodec Studio | 模块化的压缩/评测工作流 | 同一模块配置下是否能复现实验与解析总开销 |

这些接口来自各论文的方法描述 [29, 21, 39, 40, 42, 45, 44]。方法题名保留了作者的命名，不能据此推断统一协议下的优劣。

## 11.2　可见性分布决定重要性

训练视角的平均贡献可能低估背面、侧面薄结构或未来相机轨迹中的遮挡边界。设 $p_{\text{train}}(v)$ 与部署视角分布 $p_{\text{deploy}}(v)$ 不同，同一参数扰动的期望误差也不同：

$$D_p(\Theta, \hat{\Theta}) = \mathbb{E}_{v \sim p(v)}\, d\big(\mathcal{R}(\Theta, v), \mathcal{R}(\hat{\Theta}, v)\big). \tag{11.2}$$

重要性不是一个与相机无关的天然标签。可用少量独立校准视角覆盖预期轨迹，并报告分布外视角的最差分位误差；这仍不保证所有新视角安全。研究假设可以是“覆盖性约束减少轮廓崩溃”，失败判据则是同码率下的持出轮廓或图像误差没有改善。

## 11.3 理论保证必须带着假设一起阅读

将渲染误差近似成参数二次型，常见形式为

$$D(\delta\Theta) \approx \delta\Theta^{\top} J_{\mathcal{R}}^{\top} W J_{\mathcal{R}} \delta\Theta. \tag{11.3}$$

这个式子忽略了排序离散变化、可见性切换与较大的扰动，所以即使局部代理量很小，也不能直接宣称剪枝在所有相机上有误差界。REFINE 使用与渲染相关的解析近似重要性来避免直接逐基元重渲染评分，应检查代理近似与完整删除损失的相关性，而不是把它泛称为一个无条件可靠的 predictor[34]。

Provable Pruning 的 coreset 讨论明确依赖问题约束：无约束的 3DGS 不能一般性地获得非平凡的乘法 coreset。阅读其保证时必须携带指定分辨率、视图集合、有效性与对数透射率稳定性条件 [38]。将一个有限视图定理推广成连续相机空间的保证，需要额外分析。验证时可以先逐项破坏假设：制造极薄遮挡物、极高 opacity 或新视角，观察界或代理从哪里失效。

## 11.4 2025–2026 工作的可核验观察窗口

下表只列出与本书缺口直接相关、此次核查取得官方元数据及论文证据的研究方向。预印本的存在、题名和版本可核验，并不意味着本文复现了其实验或确认其会议状态。

**表 11.2** 近期工作对应的研究问题，而非性能排行榜。

| 工作 | 可检查的机制 | 建议独立验证的边界 |
|---|---|---|
| Noise-substituted VQ | 以噪声替代辅助离散优化 | 训练代理与实际码字的误差是否一致 |
| Image-Conditioned Quantization | 图像条件进入量化/解码 | 接收端是否需要额外图像；其字节与计算是否计入 |
| Scene-Adaptive Lattice VQ | 用场景自适应格量化组织码字 | 结构约束的收益能否覆盖侧信息与解码复杂度 |
| GETA-3DGS | 联合结构剪枝和量化 | 同预算下联合方法是否优于分阶段与随机预算控制 |
| KISS-GS / Non-Uniform Quantisation | 简化压缩流程与非均匀属性量化 | 元数据、属性域和不同场景的低码率失败 |
| QIRF | 非正交函数空间的表示冗余 | 函数空间代理与持出图像误差的对应；不推断量子硬件优势 |

对应论文见 [30, 43, 49, 36, 50, 51, 35]。综述可用于寻找遗漏的假设与分母，而不应代替对具体方法的核验 [28, 52, 53]。

## 11.5 三个有明确失败标准的项目

### 11.5.1 视角风险感知的混合精度

在总文件字节 $B$ 的约束下选择属性组位宽 $b_g$：

$$\min_{\{b_g\}} \sum_g w_g D_g(b_g), \qquad \sum_g n_g b_g + B_{\text{side}}(\{b_g\}) \le 8B. \tag{11.4}$$

式中 $B_{\text{side}}$ 的单位为 bit，包含分组表、量化范围和码本等。该可分解目标只是近似：不同属性误差可能相互补偿，排序变化也可能破坏局部可加性。最小实验比较统一位宽、按属性位宽、随机分配和风险感知分配，固定实有字节、同一 renderer 和相同恢复预算。若持出视角的轮廓误差及图像质量没有改善，或收益小于额外分析成本，假设就没有获得支持。

### 11.5.2 带旋转等价性的残差码本

对单位四元数考虑 $q$ 与 $-q$ 表示相同旋转；对于近各向同性的高斯，部分旋转变化几乎不改变协方差。因此旋转角误差与渲染误差不是一一对应。实验可比较归一化后的欧氏 VQ、属性分离码本和 anchor 局部切空间残差，在相同总字节下同时报告旋转、投影椭圆和图像误差。若算法只改善角度指标却恶化图像，或小数据集中的提升来自更多恢复训练，就需要重新解释贡献。

### 11.5.3 能随机访问的渐进场景

基础层保证可辨认的视角，增强层恢复局部细节。需明确每层依赖、相机请求、缓存策略和累计字节；只有传输完整低层才能解码高层时，不能将高层单独宣称为可随机访问。对同一相机轨迹比较不可恢复剪枝、几何 LOD 和渐进码流，测首帧时延、每次视角切换的额外字节、目标质量达到时间及缓存峰值。每可见区域每秒的 bit 数只有在可见区域定义、面积归一化及观察时间窗固定后才有意义。

## 11.6 动态场景、任务与其他分册的连接

给高斯增加时间条件，不自动得到动作条件的世界模型。动态表示回答“这个时刻的外观或几何是什么”，闭环模型还要回答“执行某动作后状态如何变化，以及不确定性怎样影响决策”。本书保留动态数据关联、时间残差与随机时间访问的接口；动作条件动力学与规划参见《世界模型》，控制与机器人传感器闭环参见《具身智能》。

当部署任务包含深度、语义或跟踪时，可以建立

$$D_{\text{task}} = \lambda_{\text{rgb}} D_{\text{rgb}} + \lambda_{\text{geo}} D_{\text{geo}} + \lambda_{\text{sem}} D_{\text{sem}} + \lambda_{\text{time}} D_{\text{time}}. \tag{11.5}$$

每项应有测量定义、量纲或归一化方式。调权重所用的数据不是最终测试集。选择任务指标是一个研究设计决定，不能因为某个 PSNR 更高就推断它更适合控制。

**练习与提示**

从本章选一个项目，用一页写明输入、输出、控制变量、计费边界和失败判据。再选一个可能改变结论的反例：相机分布移动、共享模型不在接收端，或高 opacity 薄结构。先设计能暴露该反例的实验，再写方法的成功故事。

# 第 12 章　从最小验证到研究原型

复杂训练系统最容易掩盖基础错误。一个图像损失可能在错误的颜色空间下降，一个稳定的四元数数组可能没有真正描述单位旋转，一个很小的压缩文件也可能依赖未计费的内存数据。本章把这些问题拆成三项可以在 CPU 上执行的验证，并给出走向真实场景实验的下一步。所有本次实测来自随书脚本，教学数据不冒充训练数据或论文基准。

## 12.1　单射线：让背景项接受手算检查

考虑两个不等宽区间，长度分别为 $\delta_1 = 1$、$\delta_2 = 1/2$，令密度为 $\sigma_1 = \log 2$、$\sigma_2 = 2\log 4$，颜色分别是红色 $(1,0,0)$ 与蓝色 $(0,0,1)$，背景为白色 $(1,1,1)$。由光学厚度可得

$$\alpha_1 = 1 - e^{-\log 2} = \frac{1}{2}, \qquad \alpha_2 = 1 - e^{-\log 4} = \frac{3}{4}, \tag{12.1}$$

$$T_1 = 1, \qquad T_2 = 1 - \alpha_1 = \frac{1}{2}, \tag{12.2}$$

$$w_1 = \frac{1}{2}, \qquad w_2 = \frac{1}{2}\cdot\frac{3}{4} = \frac{3}{8}, \tag{12.3}$$

$$T_{\mathrm{bg}} = (1-\alpha_1)(1-\alpha_2) = \frac{1}{8}. \tag{12.4}$$

因此

$$\hat{\mathbf{c}} = \frac{1}{2}(1,0,0) + \frac{3}{8}(0,0,1) + \frac{1}{8}(1,1,1) = (\tfrac{5}{8}, \tfrac{1}{8}, \tfrac{1}{2}). \tag{12.5}$$

这里的参考值来自有理数手算，独立于向量化实现。还应验证总权重 $w_1 + w_2 + T_{\mathrm{bg}} = 1$。若将 $T_2$ 错当最终背景透射率，背景会被重复计入；如果默认所有 $\delta$ 相同，也会在本例的第二段暴露错误。

随书测试还检查零密度回到背景、极大光学厚度的稳定性、空射线、常量区间细分和非法输入；光栅器另用单基元中心像素核对 alpha 与背景贡献。本次没有给多基元深度排序建立独立参考断言，不能把已有合成测试称为排序正确性的证明。本例在统一约定的数值颜色域中检查合成公式的代数正确性。

## 12.2　投影：解析 Jacobian 与独立数值差分

投影函数为 $\pi(X,Y,Z) = (f_x X/Z + c_x, f_y Y/Z + c_y)$，其导数必须使用相机坐标。先把世界协方差变换成 $R_{cw}\Sigma_w R_{cw}^{\mathsf{T}}$，再用投影 Jacobian，与先在世界坐标中计算 $J_\pi R_{cw}$ 的结果应一致：

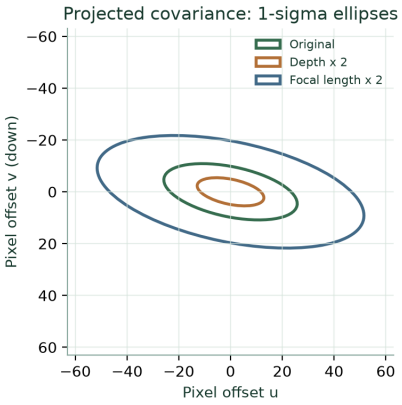
$$\Sigma_{2D} = J_\pi \left(R_{cw}\Sigma_w R_{cw}^{\mathsf{T}}\right) J_\pi^{\mathsf{T}}. \tag{12.6}$$

对世界坐标均值定义完整投影函数，再对其使用中心差分：

$$\begin{aligned} \Pi_w(\boldsymbol{\mu}_w) &= \pi(R_{cw}\boldsymbol{\mu}_w + \mathbf{t}_{cw}), \\ \frac{\Pi_w(\boldsymbol{\mu}_w + \epsilon\mathbf{e}_k) - \Pi_w(\boldsymbol{\mu}_w - \epsilon\mathbf{e}_k)}{2\epsilon} &\simeq J_\pi R_{cw}\mathbf{e}_k. \end{aligned} \tag{12.7}$$

其中 $\mathbf{e}_k$ 是世界坐标的第 $k$ 个单位基向量，$J_\pi$ 在相机均值处计算。这独立检验世界 Jacobian 的各列及外参方向，不能把解析式复制一遍充当“参考实现”。步长太小会受浮点消减影响，太大则混入投影非线性；脚本实际检查 $\epsilon = 10^{-4}, 10^{-5}, 10^{-6}, 10^{-7}$，各步长误差保存在结果文件中，避免仅凭单个差分阈值判断。

实验使用非恒等相机旋转，并检查对称性、正定性、焦距变化以及深度变化。在保持世界协方差固定的条件下，焦距乘二使计算出的未低通二维协方差乘四；将整个相机均值向量乘二、即沿同一视线把距离加倍，则使该协方差变为四分之一。对离轴点，若只改变 $Z$ 而保持 $X, Y$ 不动，Jacobian 中的 $X/Z^2, Y/Z^2$ 项有不同缩放规律，不能无条件断言整个协方差都按同一倍数变化。这些比例针对未加固定屏幕低通量的局部投影公式，不是对任意大基元的精确透视保证。本次教学场景的中心深度守卫为 $z_c \le 0.1 + 10^{-12}$ 时拒绝投影，其中 $10^{-12}$ 是用于相机变换浮点误差的绝对容差。这只检查中心；代码没有完成整个高斯支持域跨越近裁面的检测与裁切。大基元仍需按第 4 章的线性化有效条件另行检查。




```
World-to-camera convention: p_c = R_cw p_w + t_cw

Analytic world Jacobian J_camera R_cw:
   155.49606     2.49782   -14.16582
     1.21707   137.02432    19.54883

Central difference (epsilon = 1e-6):
   155.49606     2.49782   -14.16582
     1.21707   137.02432    19.54883

Max absolute error: 1.77e-08 pixel / world unit

Eigenvalues: 93.50648, 689.15743
Depth x 2 -> covariance / 4; focal x 2 -> covariance x 4
Near-plane centers rejected; nonidentity R_cw tested.
```

**图 12.1** 本次合成教学实验的投影检查。椭圆用于观察协方差与相机变化；数值误差及通过条件见实验记录。

## 12.3 固定高斯场景的压缩与剪枝 smoke test

本主题没有可用的训练 checkpoint，因此脚本生成一个可重建、可哈希的合成高斯集合。它具有已知位置、合法旋转、正尺度、不透明度和固定颜色，使用固定相机集合与 CPU 投影、深度排序及前到后合成。这是检查场景编码机制的教学场景，没有执行 COLMAP、真实图像拟合或 densification。颜色采用简化配置，不由此宣称覆盖完整高阶 SH 的复杂视角外观。

实验先冻结场景与相机，再分别编码未压缩属性、不同位宽属性、剪枝结果和组合配置。真实码流需写入磁盘，再由只读取码流的解码路径恢复表示；原场景的 min/max、码本或被删属性若仍被解码器偷偷引用，往返实验就不成立。码流字节取文件实际长度，量化范围、属性头部和打包填充一并计入。

**表 12.1** 本次合成高斯场景的真实码流与压缩失真。解码为内存码流的 CPU 墙钟中位数/p95（ms），101 次采样；Float32 自比较为精确一致。

| 方案 | 高斯数 | 总字节 | PSNR/dB | SSIM | 解码 ms |
|---|---|---|---|---|---|
| Float32 | 544 | 30,823 | ∞ | 1.0000 | 0.020/0.021 |
| 8 bit | 544 | 8,598 | 59.18 | 0.9999 | 0.035/0.036 |
| 4 bit | 544 | 4,790 | 36.35 | 0.9877 | 0.039/0.043 |
| 剪枝 25% | 408 | 23,207 | 30.97 | 0.9691 | 0.017/0.018 |
| 剪枝 + 4 bit | 408 | 3,838 | 30.40 | 0.9590 | 0.033/0.034 |

这里的 PSNR 衡量对未压缩模型图像的保真度，没有真实照片作为 ground truth。如果使用 SSIM，则按实验记录中的窗口与实现计算；没有安装 LPIPS 网络、没有真实深度与任务标签的项目

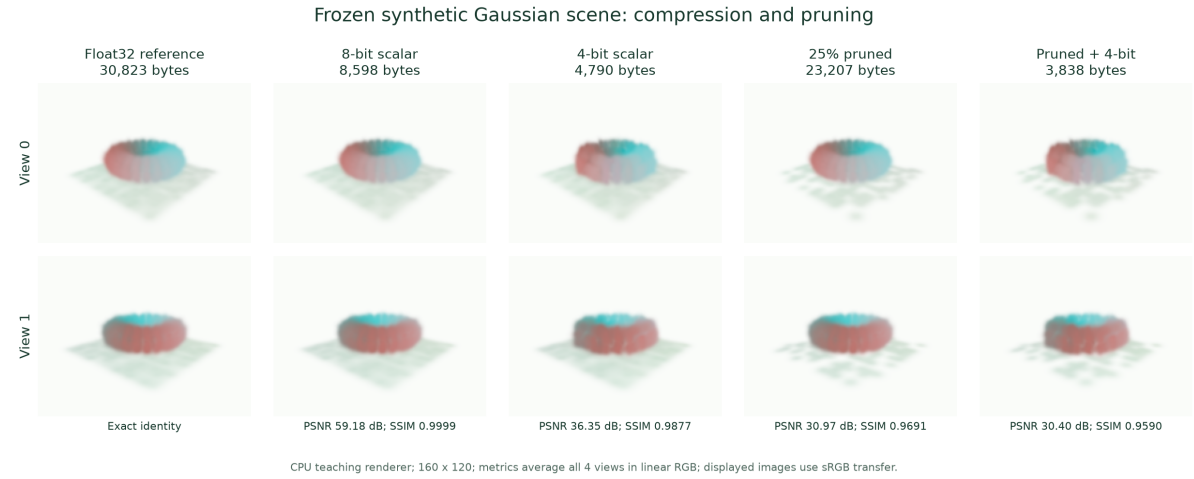



**图 12.2** 固定相机下的合成场景对照。图像用于观察位置偏移、颜色改变或剪枝缺失；画质参照为同一冻结场景的未压缩渲染。

保持未测。量化方案的编码计时包含冻结属性的范围统计、量化、打包与元数据序列化，Float32 基线则直接序列化属性；两者均不含场景生成、剪枝或写盘。内存解码与文件读取加解码分别计时。每项先预热一次，编码重复 21 次、内存解码 101 次、文件读取加解码 31 次；后者的文件可能命中系统缓存，不能解释成冷盘 I/O。四视角渲染重复 7 次，不含 PNG 保存和指标计算。这些 CPU 墙钟计时用于追踪本脚本成本，不能与 CUDA tile rasterizer 的 FPS 横向比较。操作系统调度和缓存会影响小样本计时，结果只代表本次环境和配置。

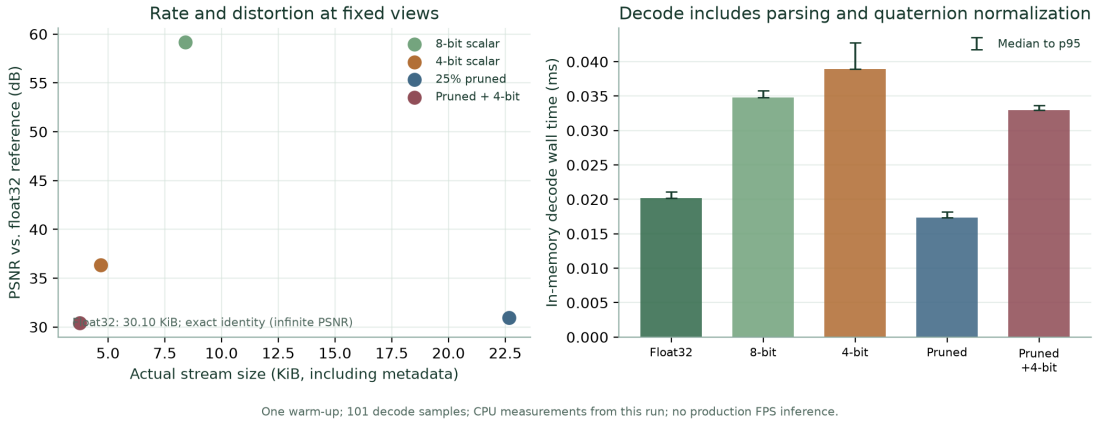



**图 12.3** 合成场景的实有码流大小与图像保真度。每个点是一个固定配置的本次实测，不代表真实场景数据集的普遍压缩曲线。

## 12.4 运行与复查记录

在本书主题目录运行下列命令；Python 环境需要 NumPy、Pillow，生成图还需要 matplotlib：

**Listing 12.1** 重跑最小实验及生成书稿。

```
python3 manuscript/experiments/run_experiments.py
bash manuscript/build.sh
```

仅执行数值与 codec 检查可使用 `--skip-plots`。输出目录为 `output/experiments/`，其中 `results.json` 保存机器可读结果，实验报告.md 给出人读说明，代码与版本信息也在同目录记录。要重现一项结果，先核对配置、输入哈希与软件版本，再检查原始逐次时间样本和测量边界；不要只复制汇总表。

**定义与约定**

本次以有理数手算参考检查射线合成，以有限差分检查投影导数，以独立码流往返检查序列化与解码，并检查所列边界案例与非法输入。多基元深度排序尚无独立参考断言；近裁面检查仅覆盖高斯中心，整个支持域的跨面检测与裁切尚未实现。

## 12.5 从合成场景走向真实重建

下一步选用许可和数据可获取性明确的小型已训练场景，保存图像、相机和 checkpoint 的哈希。训练使用原始方法或固定提交的公开实现，记录实际训练步数与硬件；校准视角只用于量化范围、码本与剪枝阈值，最终测试视角直到配置固定后才评估。先重建零压缩差异基线，检查导入/导出是否改变 SH 次序、旋转顺序或背景。

将至少一组场景细节分成三类：薄结构、低纹理遮挡物与视角相关反射。固定 renderer 后依次只量化位置、尺度、旋转、opacity 和 SH，再做剪枝与联合方法，记录各自的误差热图与实际字节。无恢复与有恢复两种预算分开比较。码流若跨场景共享模型，额外报告模型本身及摊销协议。

真实场景上的必测对象包括逐图 PSNR/SSIM/LPIPS、每场景结果、编码准备及位流生成耗时、解码延迟、CPU 内存、显存峰值和同步渲染时间。没有几何真值时，不用外观好看替代深度指标；没有 GPU 环境时，不把 CPU 教学渲染时间换算成生产性能。本次交付的合成实验结果应作为这项扩展的起点，而不是已经完成的真实复现。

## 12.6 循序开展的学习计划

**表 12.2** 按可验证产出推进，而非按固定周数追赶方法名称。

| 阶段 | 阅读与实现 | 进入下一阶段前的证据 |
|---|---|---|
| 相机与体渲染 | 第 1–2 章；射线、采样、背景 | 手算与数值一致；坐标重投影通过；解释有限末段 |
| 连续场方法 | 第 3 章；表示与采样替换 | 相同视角和预算；区分质量、抗混叠、速度与泛化条件 |
| 高斯渲染 | 第 4–5 章；投影与容量控制 | Jacobian 独立差分通过；解释排序和可见性边界 |
| 属性编码 | 第 6–7 章；scalar 与 VQ | 真实码流往返；字节账本；合法旋转与正尺度 |
| 剪枝与熵编码 | 第 8–9 章；预算与解码依赖 | 背面/遮挡反例；累计码长；侧信息计费 |

| 阶段 | 阅读与实现 | 进入下一阶段前的证据 |
| --- | --- | --- |
| 研究原型 | 第 10–12 章；配对消融 | 持出结果、失败案例、成本曲线及可复现命令 |

**练习与提示**

1. 将单射线的白背景改为黑背景，先手算再修改配置。解释为何每个通道的变化恰为 $T_{\text{bg}}$。
2. 对投影实验换一个非恒等旋转，故意漏乘 $R_{cw}$。哪项独立检查会先失败？
3. 对 4 bit codec 只计 packed 属性、不计 header，再与实际文件大小比较。小场景为何更容易由元数据主导？
4. 设计一个黑色、高 opacity 的前景高斯，使低颜色幅度的剪枝分数失效。评价应固定何种背景和视角？

# 附录 A　符号、单位与实现约定

| 符号 | 本书含义与单位 |
| --- | --- |
| $\mathbf{x}_w, \mathbf{x}_c$ | 世界坐标和相机坐标；长度单位由场景归一化定义 |
| $R_{cw}, \mathbf{t}_{cw}$ | 世界到相机变换，$\mathbf{x}_c = R_{cw}\mathbf{x}_w + \mathbf{t}_{cw}$；旋转无量纲，平移为长度 |
| $\mathbf{o}, \mathbf{d}$ | 世界相机中心和单位方向；$\mathbf{o} = -R_{cw}^{\top}\mathbf{t}_{cw}$ |
| $\mathbf{r}(s), s, Z$ | $\mathbf{o} + s\mathbf{d}$；$s$ 为沿单位射线长度，$Z$ 是相机光轴深度，通常不同 |
| $K, f_x, f_y, c_x, c_y$ | 针孔内参；焦距与主点按像素计，像素中心约定显式给出 |
| $\sigma_k, \delta_k, \alpha_k$ | 体密度为长度的倒数，区间长度为长度，体渲染区间不透明度 $\alpha_k = 1 - \exp(-\sigma_k\delta_k)$ 无量纲 |
| $T, w, T_{\text{bg}}$ | 无量纲透射率、样本合成权重和尾部背景权重 |
| $\boldsymbol{\mu}, \Sigma$ | 高斯均值与三维协方差；单位分别为长度及长度平方 |
| $\mathbf{q}, \mathbf{s}$ | 单位四元数与正标准差；本书采用实部在前的 $(w, x, y, z)^{\top}$，$\mathbf{q}$ 与 $-\mathbf{q}$ 表示同一旋转 |
| $J_\pi, \Sigma_{2D}$ | 相机投影的导数，单位为像素/长度；二维协方差为像素平方 |
| $o_i \equiv \alpha_i$ | 高斯基础不透明度；第 4 章用 $\alpha_i$，压缩章节也用 $o_i$，二者为同一属性，不能解释为体密度 |
| $a_i(\mathbf{u})$ | 像素 $\mathbf{u}$ 上的 splat alpha，$a_i(\mathbf{u}) = o_i g_i(\mathbf{u})$；与基元基础不透明度区分 |
| $Y_{\ell m}, \mathbf{h}_{\ell m}$ | 实球谐基底及 RGB 系数；方向、归一化、顺序和色偏须与实现一致 |
| $\Delta, z, q, \hat{x}$ | 标量量化步长、整数零点、整数码与重建值；$q$ 仅在此上下文指码字 |
| $D_{\text{dist}}, R_{\text{bits}}$ | 失真与码流 bit 数；$R_{cw}$ 中的 $R$ 始终表示旋转 |
| $E, D, \mathcal{R}$ | 编码器、解码器和渲染器；与失真 $D$ 依上下文区分 |
| $N, L, M$ | 高斯数、SH 最高阶、VQ 语境中的码本大小；第 4 章复杂度中的 $M$ 则为高斯 –tile 实例数，应按上下文区分 |

## A.1 常用公式与量纲检查

$$T(s) = \exp\left[-\int_{s_n}^{s} \sigma(\mathbf{r}(\xi))\, d\xi\right], \tag{A.1}$$

$$\hat{\mathbf{c}} = \sum_k \left(\alpha_k \prod_{j<k}(1-\alpha_j)\right)\mathbf{c}_k + \left(\prod_k (1-\alpha_k)\right)\mathbf{c}_{\text{bg}}, \tag{A.2}$$

$$\Sigma_w = R(\mathbf{q})\,\mathrm{diag}(s_1^2, s_2^2, s_3^2)R(\mathbf{q})^\top, \tag{A.3}$$

$$\Sigma_{2D} \approx J_\pi R_{cw} \Sigma_w R_{cw}^\top J_\pi^\top + \epsilon I_2. \tag{A.4}$$

这里第二式的 $\alpha_k$ 是体渲染区间不透明度；用于高斯逐像素合成时，应代入 $a_i(\mathbf{u})$，不能把基础属性 $o_i \equiv \alpha_i$ 直接当成所有像素上的 alpha。最后一式的 $\epsilon$ 具有像素平方单位；它是屏幕数值/过滤设置，不是可以随意当作世界三维尺度的无量纲常量。相机旋转应满足 $R^\top R = I$、$\det R = 1$；投影正定性还要求高斯协方差合法、点处于有效相机区域且投影导数有秩。

## A.2 成本量级不等于实测性能

粗略的 NeRF 查询量为 $HWK$，还应乘网络或插值的每次查询成本。高斯渲染需同时考虑可见基元、tile 重复、排序工作区与像素覆盖，不能只用 $N$ 估计 FPS。属性数组估算

$$M_{\text{attr}} = \sum_a N_a n_a b_a / 8 \tag{A.5}$$

只计属性净荷；磁盘码流还要加 header、模型、码本、索引、填充等，训练显存还要加梯度、优化器状态与临时缓冲。十进制 MB 为 $10^6$ byte，MiB 为 $2^{20}$ byte，单位转换应明确。

# 附录 B　术语与代表练习提示

**辐射场 / NeRF**　从位置和方向到密度与颜色的连续表示；原始 NeRF 的密度分支不依赖观察方向。

**高斯泼溅 / splatting**　将三维基元的投影核累积到像素；标准 3DGS 的屏幕 alpha 合成使用近似的深度次序。

**不透明度 / opacity**　概率样的无量纲合成量；在体渲染中由密度和区间推导，在 splat 中由基元属性和屏幕核决定。

**球谐 / SH**　方向函数的基展开；阶数截断、系数量化、方向取反与基底次序是不同操作。

**容量控制 / densification**　clone、split 和 pruning 等改变基元配置的操作；替换参数时应同步优化器状态。

**PTQ / QAT**　训练后量化和量化感知训练；后者需额外训练成本，STE 只是一种梯度代理。

**VQ / 残差 VQ**　用码本索引替代属性向量；多个残差阶段需计入每阶段码本和索引。

**率失真**　给定失真定义与成本边界，在码率与误差间选择表示；实有字节不同于理想熵估计。

**校准 / 测试**　校准用于范围、码本、阈值或预算选择；最终测试用于配置固定后的评价。

**LOD / 渐进码流**　前者选择不同细节表示，后者规定可解码的层或前缀；只有满足数据依赖时才可随机访问。

**冷启动 / 稳态**　冷启动包括文件、解码、上传与首帧，稳态计时固定已准备好的场景及相机轨迹。

## B.1　检查答案时应抓住的依据

**背景与最后区间**。空射线的颜色就是背景，权重总和在加上尾透射率后为一。射线区间的末端必须来自明确的 far bound 或模型背景策略，不能因为实现方便附加无限长区间并仍宣称是有限段积分。

**投影与坐标**。单位高斯在相机坐标旋转后仍须通过同一外参投影；先变换协方差再乘 Jacobian，与对世界坐标求导的 $J_\pi R_{cw}$ 应相容。忽略旋转、对方向加平移、把相机 $Z$ 当单位射线距离，都会破坏这些检查。

**删除与遮挡**。黑色前景也能通过遮挡白背景产生重大影响。删除损失增量定义为删除后的损失减去删除前损失；它可为负，表示原来基元确实损害当前目标，但这不构成所有视角上的保证。多个删除操作之间有交互，不能把单点分数简单相加当作精确总损失。

**量化与真实成本**。四元数解码后的单位化不消除所有方向误差；零向量需要显式处理，$q$ 与 $-q$ 的符号规范应在码本学习前一致。尺度在 log 域量化后直接取 exp 保持正值，不减去不明常数。小场景的 metadata、码本和填充可能超过属性净荷。

**两个 PSNR 平均**。对第 10 章给出的 $10^{-2}, 10^{-4}$ 两项 MSE，逐图 PSNR 为 20、40 dB，均值为 30 dB；总体平均 MSE 为 0.00505，相应 PSNR 约为 22.97 dB。两种结果都可计算，但回答的是不同聚合问题。

**论文主张与本书实测**。本书合成 codec 的图像参照是未压缩模型输出，不是真实照片。如果代码未训练场景、未运行 GPU、未加载 LPIPS 权重，就分别不能给出训练质量、生产 FPS 或 LPIPS 分数。将未运行的计划写清楚，比用不相干数字填满表格更有用。

# 参考文献与证据说明

本书引用用于支持具体模型、算法及适用条件，不组成跨论文性能排行榜。参考文献的年份采用被核查的 arXiv 首发年份，已确认的会议或期刊信息另记；题名中的性能数字是原题名的一部分，不是本次复现结果。`evidence/bibliography-audit.json` 保存逐条核查的来源、版本、日期与状态，`evidence/citation-claims.md` 记录引文支持的论断及边界。后续版本可能变动，复现实验应锁定论文版本、实现提交与评测配置。本书自己的合成实验见第 12 章，不能据此推断真实场景的训练速度或几何精度。


[1] OpenCV Contributors. *Camera Calibration and 3D Reconstruction: calib3d Module Documentation*. OpenCV 4.x source documentation, revision 87eff0a6 (2026-09-13); projection equations checked 2026-10-22. Sept. 13, 2026. URL: https://github.com/opencv/opencv/blob/87eff0a6d2aa2d7187775b964565ba32b786673f/modules/calib3d/include/opencv2/calib3d.hpp (visited on 10/22/2026).

[2] W3C CSS Working Group. *CSS Color Module Level 4*. Candidate Recommendation Draft, 2026-10-20; sRGB transfer functions checked 2026-10-22. Oct. 20, 2026. URL: https://www.w3.org/TR/2026/CRD-css-color-4-20261020/ (visited on 10/22/2026).

[3] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. *NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis*. arXiv v2 (2020-08-03); metadata checked 2026-10-21. 2020. arXiv: 2003.08934. URL: https://arxiv.org/abs/2003.08934v2 (visited on 10/21/2026).

[4] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan. *Mip-NeRF: A Multiscale Representation for Anti-Aliasing Neural Radiance Fields*. arXiv v3 (2021-08-14); metadata checked 2026-10-21. 2021. arXiv: 2103.13415. URL: https://arxiv.org/abs/2103.13415v3 (visited on 10/21/2026).

[5] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman. *Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields*. arXiv v3 (2022-03-25); metadata checked 2026-10-21. 2021. arXiv: 2111.12077. URL: https://arxiv.org/abs/2111.12077v3 (visited on 10/21/2026).

[6] T. Müller, A. Evans, C. Schied, and A. Keller. *Instant Neural Graphics Primitives with a Multiresolution Hash Encoding*. arXiv v2 (2022-05-04); metadata checked 2026-10-21. 2022. DOI: 10.1145/3528223.3530127. arXiv: 2201.05989. URL: https://arxiv.org/abs/2201.05989v2 (visited on 10/21/2026).

[7] C. Sun, M. Sun, and H.-T. Chen. *Direct Voxel Grid Optimization: Super-fast Convergence for Radiance Fields Reconstruction*. arXiv v2 (2022-06-03); metadata checked 2026-10-21. 2021. arXiv: 2111.11215. URL: https://arxiv.org/abs/2111.11215v2 (visited on 10/21/2026).

[8] A. Yu, S. Fridovich-Keil, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa. *Plenoxels: Radiance Fields without Neural Networks*. arXiv v1 (2021-12-09); metadata checked 2026-10-21. 2021. arXiv: 2112.05131. URL: https://arxiv.org/abs/2112.05131v1 (visited on 10/21/2026).

[9] A. Chen, Z. Xu, A. Geiger, J. Yu, and H. Su. *TensoRF: Tensorial Radiance Fields*. arXiv v2 (2022-11-29); metadata checked 2026-10-21. 2022. arXiv: 2203.09517. URL: https://arxiv.org/abs/2203.09517v2 (visited on 10/21/2026).

[10] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman. *Zip-NeRF: Anti-Aliased Grid-Based Neural Radiance Fields*. arXiv v3 (2023-10-26); metadata checked 2026-10-21. 2023. arXiv: 2304.06706. URL: https://arxiv.org/abs/2304.06706v3 (visited on 10/21/2026).

[11] R. Li, H. Gao, M. Tancik, and A. Kanazawa. *NerfAcc: Efficient Sampling Accelerates NeRFs*. arXiv v2 (2023-10-24); metadata checked 2026-10-21. 2023. arXiv: 2305.04966. URL: https://arxiv.org/abs/2305.04966v2 (visited on 10/21/2026).

[12] A. Yu, V. Ye, M. Tancik, and A. Kanazawa. *pixelNeRF: Neural Radiance Fields from One or Few Images*. arXiv v3 (2021-05-30); metadata checked 2026-10-21. 2020. arXiv: 2012.02190. URL: https://arxiv.org/abs/2012.02190v3 (visited on 10/21/2026).

[13] Q. Wang, Z. Wang, K. Genova, P. Srinivasan, H. Zhou, J. T. Barron, R. Martin-Brualla, N. Snavely, and T. Funkhouser. *IBRNet: Learning Multi-View Image-Based Rendering*. arXiv v2 (2021-04-06); metadata checked 2026-10-21. 2021. arXiv: 2102.13090. URL: https://arxiv.org/abs/2102.13090v2 (visited on 10/21/2026).

[14] M. Niemeyer, J. T. Barron, B. Mildenhall, M. S. M. Sajjadi, A. Geiger, and N. Radwan. *RegNeRF: Regularizing Neural Radiance Fields for View Synthesis from Sparse Inputs*. arXiv v1 (2021-12-01); metadata checked 2026-10-21. 2021. arXiv: 2112.00724. URL: https://arxiv.org/abs/2112.00724v1 (visited on 10/21/2026).

[15] R. Martin-Brualla, N. Radwan, M. S. M. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. Duckworth. *NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections*. arXiv v3 (2021-01-06); metadata checked 2026-10-21. 2020. arXiv: 2008.02268. URL: https://arxiv.org/abs/2008.02268v3 (visited on 10/21/2026).

[16] C.-H. Lin, W.-C. Ma, A. Torralba, and S. Lucey. *BARF: Bundle-Adjusting Neural Radiance Fields*. arXiv v2 (2021-08-19); metadata checked 2026-10-21. 2021. arXiv: 2104.06405. URL: https://arxiv.org/abs/2104.06405v2 (visited on 10/21/2026).

[17] P. Wang, L. Liu, Y. Liu, C. Theobalt, T. Komura, and W. Wang. *NeuS: Learning Neural Implicit Surfaces by Volume Rendering for Multi-view Reconstruction*. arXiv v3 (2023-02-01); metadata checked 2026-10-22. 2021. arXiv: 2106.10689. URL: https://arxiv.org/abs/2106.10689v3 (visited on 10/22/2026).

[18] A. Pumarola, E. Corona, G. Pons-Moll, and F. Moreno-Noguer. *D-NeRF: Neural Radiance Fields for Dynamic Scenes*. arXiv v1 (2020-11-27); metadata checked 2026-10-22. 2020. arXiv: 2011.13961. URL: https://arxiv.org/abs/2011.13961v1 (visited on 10/22/2026).

[19] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis. *3D Gaussian Splatting for Real-Time Radiance Field Rendering*. arXiv v1 (2023-08-08); metadata checked 2026-10-21. 2023. arXiv: 2308.04079. URL: https://arxiv.org/abs/2308.04079v1 (visited on 10/21/2026).

[20] Z. Yu, A. Chen, B. Huang, T. Sattler, and A. Geiger. *Mip-Splatting: Alias-free 3D Gaussian Splatting*. arXiv v1 (2023-11-27); metadata checked 2026-10-21. 2023. arXiv: 2311.16493. URL: https://arxiv.org/abs/2311.16493v1 (visited on 10/21/2026).

[21] T. Lu, M. Yu, L. Xu, Y. Xiangli, L. Wang, D. Lin, and B. Dai. *Scaffold-GS: Structured 3D Gaussians for View-Adaptive Rendering*. arXiv v1 (2023-11-30); metadata checked 2026-10-21. 2023. arXiv: 2312.00109. URL: https://arxiv.org/abs/2312.00109v1 (visited on 10/21/2026).

[22] Q. Feng, G. Cao, H. Chen, T.-J. Mu, R. R. Martin, and S.-M. Hu. *A New Split Algorithm for 3D Gaussian Splatting*. arXiv v1 (2024-03-14); metadata checked 2026-10-21. 2024. arXiv: 2403.09143. URL: https://arxiv.org/abs/2403.09143v1 (visited on 10/21/2026).

[23] Y. Chen, H. Xu, C. Zheng, B. Zhuang, M. Pollefeys, A. Geiger, T.-J. Cham, and J. Cai. *MVSplat: Efficient 3D Gaussian Splatting from Sparse Multi-View Images*. arXiv v2 (2024-07-18); metadata checked 2026-10-22. 2024. DOI: 10.1007/978-3-031-72664-4_21. arXiv: 2403.14627. URL: https://arxiv.org/abs/2403.14627v2 (visited on 10/22/2026).

[24] B. Huang, Z. Yu, A. Chen, A. Geiger, and S. Gao. *2D Gaussian Splatting for Geometrically Accurate Radiance Fields*. arXiv v3 (2025-02-22); metadata checked 2026-10-21. 2024. DOI: 10.1145/3641519.3657428. arXiv: 2403.17888. URL: https://arxiv.org/abs/2403.17888v3 (visited on 10/21/2026).

[25] S. Kheradmand, D. Rebain, G. Sharma, W. Sun, J. Tseng, H. Isack, A. Kar, A. Tagliasacchi, and K. M. Yi. *3D Gaussian Splatting as Markov Chain Monte Carlo*. arXiv v3 (2025-02-12); metadata checked 2026-10-21. 2024. arXiv: 2404.09591. URL: https://arxiv.org/abs/2404.09591v3 (visited on 10/21/2026).

[26] G. Wu, T. Yi, J. Fang, L. Xie, X. Zhang, W. Wei, W. Liu, Q. Tian, and X. Wang. *4D Gaussian Splatting for Real-Time Dynamic Scene Rendering*. arXiv v3 (2024-07-15); metadata checked 2026-10-22. 2023. arXiv: 2310.08528. URL: https://arxiv.org/abs/2310.08528v3 (visited on 10/22/2026).

[27] J. Luiten, G. Kopanas, B. Leibe, and D. Ramanan. *Dynamic 3D Gaussians: Tracking by Persistent Dynamic View Synthesis*. arXiv v1 (2023-08-18); metadata checked 2026-10-22. 2023. arXiv: 2308.09713. URL: https://arxiv.org/abs/2308.09713v1 (visited on 10/22/2026).

[28] M. T. Bagdasarian, P. Knoll, Y.-H. Li, F. Barthel, A. Hilsmann, P. Eisert, and W. Morgenstern. *3DGS.zip: A survey on 3D Gaussian Splatting Compression Methods*. arXiv v5 (2025-03-05); metadata checked 2026-10-21. 2024. DOI: 10.1111/cgf.70078. arXiv: 2407.09510. URL: https://arxiv.org/abs/2407.09510v5 (visited on 10/21/2026).

[29] Z. Fan, K. Wang, K. Wen, Z. Zhu, D. Xu, and Z. Wang. *LightGaussian: Unbounded 3D Gaussian Compression with 15x Reduction and 200+ FPS*. arXiv v6 (2024-11-12); metadata checked 2026-10-21. 2023. arXiv: 2311.17245. URL: https://arxiv.org/abs/2311.17245v6 (visited on 10/21/2026).

[30] H. Wang, M. H. Vali, and A. Solin. *Compressing 3D Gaussian Splatting by Noise-Substituted Vector Quantization*. arXiv v2 (2025-04-08); metadata checked 2026-10-21. 2025. DOI: 10.1007/978-3-031-95911-0_24. arXiv: 2504.03059. URL: https://arxiv.org/abs/2504.03059v2 (visited on 10/21/2026).

[31] Z. Zhang, T. Song, Y. Lee, L. Yang, C. Peng, R. Chellappa, and D. Fan. *LP-3DGS: Learning to Prune 3D Gaussian Splatting*. arXiv v1 (2024-05-29); metadata checked 2026-10-21. 2024. arXiv: 2405.18784. URL: https://arxiv.org/abs/2405.18784v1 (visited on 10/21/2026).

[32] M. S. Ali, M. Qamar, S.-H. Bae, and E. Tartaglione. *Trimming the Fat: Efficient Compression of 3D Gaussian Splats through Pruning*. arXiv v2 (2024-07-29); metadata checked 2026-10-21. 2024. arXiv: 2406.18214. URL: https://arxiv.org/abs/2406.18214v2 (visited on 10/21/2026).

[33] A. N. Razlighi, E. B. Golezani, and S. Kasaei. *Confident Splatting: Confidence-Based Compression of 3D Gaussian Splatting via Learnable Beta Distributions*. arXiv v1 (2025-06-28); metadata checked 2026-10-21. 2025. arXiv: 2506.22973. URL: https://arxiv.org/abs/2506.22973v1 (visited on 10/21/2026).

[34] Z. Chen, S. Wan, M. Yu, F. Yang, and J. Hou. *REFINE: Super-efficient 3D Gaussian Splatting Pruning via Rendering-Free Primitive Importance*. arXiv v4 (2026-08-17); metadata checked 2026-10-22. 2026. arXiv: 2606.09074. URL: https://arxiv.org/abs/2606.09074v4 (visited on 10/22/2026).

[35] S. Jiang, H. Zhang, X. Ma, Y. Hu, and T. Zhang. *QIRF Quantum-Inspired Non-Orthogonal Function-Space Compression for 3D Gaussian Splatting*. arXiv v1 (2026-07-20); metadata checked 2026-10-22. 2026. arXiv: 2607.18067. URL: https://arxiv.org/abs/2607.18067v1 (visited on 10/22/2026).

[36] B. Zhang and W. Sui. *GETA-3DGS: Automatic Joint Structured Pruning and Quantization for 3D Gaussian Splatting*. arXiv v1 (2026-05-03); metadata checked 2026-10-22. 2026. arXiv: 2605.02086. URL: https://arxiv.org/abs/2605.02086v1 (visited on 10/22/2026).

[37] J. Jo, H. Kim, and J. Park. *Identifying Unnecessary 3D Gaussians using Clustering for Fast Rendering of 3D Gaussian Splatting*. arXiv v2 (2024-09-25); metadata checked 2026-10-21. 2024. arXiv: 2402.13827. URL: https://arxiv.org/abs/2402.13827v2 (visited on 10/21/2026).

[38] W. Mousa and A. Maalouf. *Provable Pruning for Efficient 3D Gaussian Splatting via Coresets*. arXiv v1 (2026-07-02); metadata checked 2026-10-22. 2026. arXiv: 2607.02721. URL: https://arxiv.org/abs/2607.02721v1 (visited on 10/22/2026).

[39] Y. Chen, Q. Wu, W. Lin, M. Harandi, and J. Cai. *HAC: Hash-grid Assisted Context for 3D Gaussian Splatting Compression*. arXiv v3 (2024-07-12); metadata checked 2026-10-21. 2024. arXiv: 2403.14530. URL: https://arxiv.org/abs/2403.14530v3 (visited on 10/21/2026).

[40] Y. Chen, Q. Wu, W. Lin, M. Harandi, and J. Cai. *HAC++: Towards 100X Compression of 3D Gaussian Splatting*. arXiv v4 (2025-02-11); metadata checked 2026-10-21. 2025. arXiv: 2501.12255. URL: https://arxiv.org/abs/2501.12255v4 (visited on 10/21/2026).

[41] L. Liu, Z. Chen, W. Jiang, W. Wang, and D. Xu. *HEMGS: A Hybrid Entropy Model for 3D Gaussian Splatting Data Compression*. arXiv v2 (2025-04-22); metadata checked 2026-10-21. 2024. arXiv: 2411.18473. URL: https://arxiv.org/abs/2411.18473v2 (visited on 10/21/2026).

[42] Y. Chen, Q. Wu, M. Li, W. Lin, M. Harandi, and J. Cai. *Fast Feedforward 3D Gaussian Splatting Compression*. arXiv v3 (2025-03-12); metadata checked 2026-10-21. 2024. arXiv: 2410.08017. URL: https://arxiv.org/abs/2410.08017v3 (visited on 10/21/2026).

[43] X. Liu, R. B. Li, K. Suzuki, and T. Nguyen. *Image-Conditioned 3D Gaussian Splat Quantization*. arXiv v2 (2025-09-30); metadata checked 2026-10-21. 2025. arXiv: 2508.15372. URL: https://arxiv.org/abs/2508.15372v2 (visited on 10/21/2026).

[44] S. Li, C. Wu, H. Li, X. Gao, Y. Liao, and L. Yu. *GSCodec Studio: A Modular Framework for Gaussian Splat Compression*. arXiv v1 (2025-06-02); metadata checked 2026-10-21. 2025. arXiv: 2506.01822. URL: https://arxiv.org/abs/2506.01822v1 (visited on 10/21/2026).

[45] Y. Chen, M. Li, Q. Wu, W. Lin, M. Harandi, and J. Cai. *PCGS: Progressive Compression of 3D Gaussian Splatting*. arXiv v1 (2025-03-11); metadata checked 2026-10-21. 2025. arXiv: 2503.08511. URL: https://arxiv.org/abs/2503.08511v1 (visited on 10/21/2026).

[46] M. Tancik, E. Weber, E. Ng, R. Li, B. Yi, J. Kerr, T. Wang, A. Kristoffersen, J. Austin, K. Salahi, A. Ahuja, D. McAllister, and A. Kanazawa. *Nerfstudio: A Modular Framework for Neural Radiance Field Development*. arXiv v4 (2023-10-17); metadata checked 2026-10-21. 2023. DOI: 10.1145/3588432.3591516. arXiv: 2302.04264. URL: https://arxiv.org/abs/2302.04264v4 (visited on 10/21/2026).

[47] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. "Image Quality Assessment: From Error Visibility to Structural Similarity". In: *IEEE Transactions on Image Processing* 13.4 (2004). DOI metadata and author-hosted original checked 2026-10-22, pp. 600–612. DOI: 10.1109/TIP.2003.819861. URL: https://doi.org/10.1109/TIP.2003.819861 (visited on 10/22/2026).

[48] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. *The Unreasonable Effectiveness of Deep Features as a Perceptual Metric*. arXiv v2 (2018-04-10); metadata checked 2026-10-22. 2018. arXiv: 1801.03924. URL: https://arxiv.org/abs/1801.03924v2 (visited on 10/22/2026).

[49] H. Xu, X. Wu, and X. Zhang. *Improving 3D Gaussian Splatting Compression by Scene-Adaptive Lattice Vector Quantization*. arXiv v2 (2026-05-20); metadata checked 2026-10-21. 2025. arXiv: 2509.13482. URL: https://arxiv.org/abs/2509.13482v2 (visited on 10/21/2026).

[50] W. Morgenstern, F. E. Branschke, F. Fleischmann, A. Szatmari, P. Schlack, F. Barthel, P. Eisert, and A. Hilsmann. *KISS-GS: 3D Gaussian Splatting Compression Kept Simple*. arXiv v1 (2026-08-27); metadata checked 2026-10-22. 2026. DOI: 10.1007/978-3-032-37461-5_37. arXiv: 2608.26948. URL: https://arxiv.org/abs/2608.26948v1 (visited on 10/22/2026).

[51] B. Van hauwermeiren, P. R. Alface, and A. Munteanu. *Non-Uniform Quantisation for 3DGS Compression*. arXiv v1 (2026-08-28); metadata checked 2026-10-22. 2026. arXiv: 2608.28272. URL: https://arxiv.org/abs/2608.28272v1 (visited on 10/22/2026).

[52] G. Chen and W. Wang. *A Survey on 3D Gaussian Splatting*. arXiv v9 (2026-04-09); metadata checked 2026-10-21. 2024. arXiv: 2401.03890. URL: https://arxiv.org/abs/2401.03890v9 (visited on 10/21/2026).

[53] T. Wu, Y.-J. Yuan, L.-X. Zhang, J. Yang, Y.-P. Cao, L.-Q. Yan, and L. Gao. *Recent Advances in 3D Gaussian Splatting*. arXiv v2 (2024-04-13); metadata checked 2026-10-21. 2024. arXiv: 2403.11134. URL: https://arxiv.org/abs/2403.11134v2 (visited on 10/21/2026).