SuGaR

论文：《SuGaR: Surface-Aligned Gaussian Splatting for Efficient 3D Mesh Reconstruction and High-Quality Mesh Rendering》
地址：https://arxiv.org/abs/2311.12775
年份：CVPR 2024

Introduction

任务：从 3D 高斯中提取 mesh
技术贡献：
(1) 使用正则项来使 gaussian 贴合物体的表面，使得 gaussian 的分布在几何上是正确的；
(2) 提取 mesh 之后，将 gaussian 绑定在 mesh 上进行优化，使得 mesh 更加准确。

Method

主要分为 3 个步骤：引入正则项优化 gaussian，使用泊松重建提取 mesh，将 gaussian 绑定在 mesh 上继续优化得到更准确的 mesh。

Aligning the Gaussians with the Surface

首先考虑一个密度函数 d:

$$d(p) = \sum_g \alpha_g \exp(-\frac{1}{2}(p-\mu_g)^T \Sigma_g^{-1}(p-\mu_g))$$

其中 p 空间中的一个点，$\mu_g, \Sigma_g, \alpha_g$ 分别表示 gaussian 的中心、协方差矩阵和不透明度。
我们想要设计一个正则项 $|d(p) - \bar{d}(p)|$，让 gaussian 能够贴合物体的表面，所以需要知道在理想情况下的 $\bar{d}(p)$ 会是什么形式。
(1) 首先是对 p 做约束，使其是接近物体表面的点，这样的话离它最近的 gaussian g^* 会对 $d(p)$ 贡献最大，且远大于其他 gaussian，可以将 $d(p)$ 近似为这一个 gaussian 的贡献：

$$\alpha_g^* \exp(-\frac{1}{2}(p-\mu_{g^*})^T \Sigma_{g^*}^{-1}(p-\mu_{g^*}))$$

(2) 然后让 3d gaussian 尽可能是一个扁平的圆盘，这样就能更好地贴合 mesh 的表面。因此就会有

$$(p-\mu_g)^T \Sigma_g^{-1}(p-\mu_g) \approx \frac{1}{s_g^2}\langle p-\mu_g, n_g\rangle^2$$

其中 s_g 和 n_g 是最短轴的长度和方向。
上式可以按下面的过程推导得到 (论文作者在 github issue 中给出的回答 https://github.com/Anttwo/SuGaR/issues/2)：

Σ_g^{-1} 可以写为 $R_g S_g^{-1} S_g^{-1} R_g^T$，其中 S_g 就是表示 scaling 的对角阵，由于现在 gaussian 是一个扁平的圆盘，因此对角线上会有一个特别小的值，即 s_g，S_g^{-1} 也会是一个对角阵，且对角线上会有两个特别小的值，还有一个值为 $\frac{1}{s_g}$，可以将这个值记为是对角线上第 i 个元素。对 $(p-\mu_g)$ 左乘 $S_g^{-1} R_g^T$ 就等价于将 $(p-\mu_g)$ 往 R_g 的第 i **列** 上做投影，并将结果做 scaling $\frac{1}{s_g}$。R_g 的第 i 列向量是 n_g，所以就可以得到上式。
(3) 最后我们希望 gaussian 要么是全透明的，要么是完全不透明的，因此设置 α_g 为 1。
如果 gaussian 能满足以上 3 点，那么 $d(p)$ 就会变为 $\bar{d}(p)$:

$$\bar{d}(p) = \exp(-\frac{1}{2s_{g^*}^2}\langle p-\mu_g^*, n_g^*\rangle^2)$$

这样的话将 $|d(p)-\bar{d}(p)|$ 加到 loss 中就能有效地约束 gaussian，使其贴合物体表面。但作者对上面推到的结果稍微转化一下，变为 SDF 相关的 loss，能有更好的效果。
想要计算 SDF，先得有距离，距离是 $|\langle p-\mu_g, n_g\rangle|$，而这刚好是上面 $\bar{d}(p)$ 中的一项:

$$\bar{d}(p) = \exp(-\frac{1}{2s_g^2}\bar{f}(p)^2)$$

因此就有:

$$\bar{f}(p) = \pm s_g^* \sqrt{-2\log(\bar{d}(p))}$$

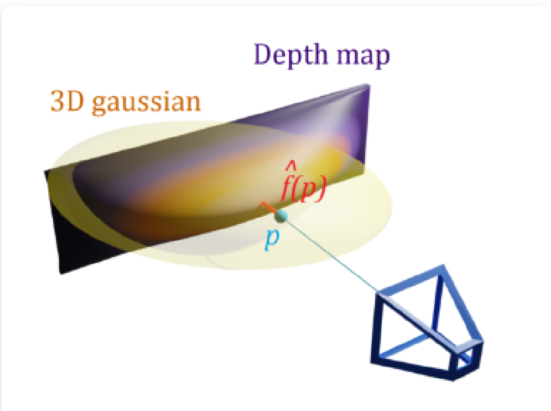
假设在理想场景中，gaussian 贴合物体表面，也就是满足上面提到的 3 个性质，那么有 $d(p)=\bar{d}(p)$，可以将上式中的 $\bar{d}(p)$ 替换为 $d(p)$:

$$f(p) = \pm s_g^* \sqrt{-2\log(d(p))}$$

我们认为 $f(p)$ 就是理想情况下的距离函数。因此设计正则项为:

$$\mathcal{R} = \frac{1}{|\mathcal{P}|}\sum_{p\in\mathcal{P}}|\hat{f}(p)-f(p)|$$

其中 $f(p)$ 是理想情况下的距离，$\hat{f}(p)$ 是估计的距离，是 p 到由 gaussian 组成的表面之间的距离。



文章说计算 $\hat{f}(p)$ 是一个 priori challenging，所以采用计算深度的方式来近似 (也是参考论文作者在 github issue 中给出的回答 https://github.com/Anttwo/SuGaR/issues/2):
可以渲染 gaussian 的深度图，而这个深度图可以近似代表场景中物体的表面位置，对于空间点中 p，在

给定视角下，可以将其投影到 2d 得到对应坐标下的深度值，这就是物体表面的深度，然后我们也知道 p 的深度，两者做差即可得到一个近似的距离。
通过在高斯分布上采样来得到 p:

$$p \sim \prod_g \mathcal{N}(.; \mu_g, \Sigma_g)$$

还添加了一项 loss 使得 f 和 $\bar{f}$ 的 normal 保持一致:

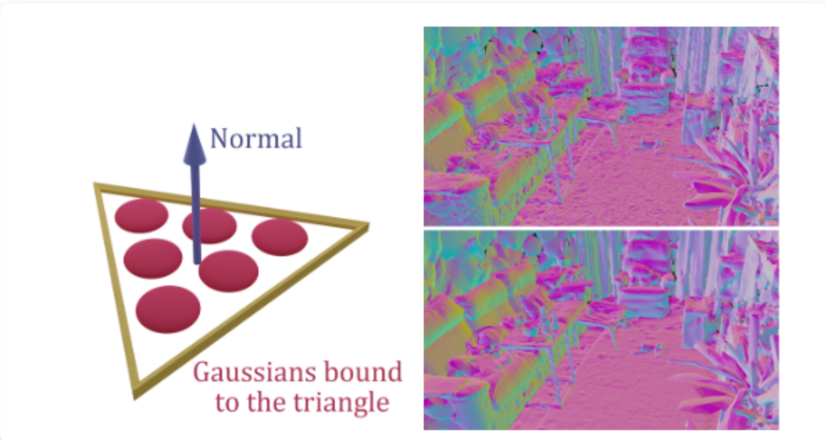
$$\mathcal{R}_{\text{Norm}} = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} || \frac{\nabla f(p)}{||\nabla f(p)||_2} - n_{g^*} ||^2$$

Efficient Mesh Extraction

TBD.
需要先了解泊松重建原理再来看这一节

Binding New 3D Gaussians to the Mesh

通过泊松重建得到 mesh 后，将 gaussian 绑定在 mesh 上进行优化来对 mesh 进行 refine。



对 mesh 上每个三角面片，初始化 n 个 gaussian，然后根据面片边长和朝向设置它们的 scale 和 rotation 等。

Experiments

训练流程：先不加入正则项训练 7000 步；然后加入一个对透明度的 entropy loss 训练 2000 步，目的是让 gaussian 变得完全不透明或者完全透明；然后将不透明度低于 0.5 的 gaussian 去除，引入正则化项训练 6000 步。这样一共训练 15000 步。
对 mesh 进行 refine 会进行 2000，7000，或 15000 步。