
计算机图形学大作业二：InfoNerf

刘泓尊 2022210866 计算机系

2022 年 11 月 24 日

基于 Jittor 复现了 InfoNerf[1]，并在 lego 模型上训练和测试。

1 原理说明

Nerf(Neural Radiance Field)[2] 是使用神经网络处理已知视角的图像信息合成新视角图像的技术。Radiance Field 可以定义为 $F : (\mathbf{x}, \mathbf{d}) \rightarrow (\mathbf{c}, \sigma)$, 其中 $\mathbf{x} \in \mathbb{R}^3$ 是位置信息, $\mathbf{d} \in \mathbb{R}^2$ 是观察方向, RF 函数输出该参数对应的点的颜色 $\mathbf{c} \in \mathbb{R}^3$ (RGB) 和体密度 σ 。为了得到渲染图像, 还需要根据该 RF 进行体渲染 (Volume Rendering) 过程。

1.1 体渲染 (Volume Rendering)

体渲染通过空间中每个点的 RGB 和体密度, 得到该视点方向上每个点对最终图像的贡献。

这一过程使用体渲染公式计算。考虑一个视线 $\mathbf{r}(t) = \mathbf{x} + t\mathbf{d}$, 在直线上一点 $\mathbf{r}(t)$ 的光强是 $\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt$, 定义从该点到该视点的衰减系数为 $T(t) = \exp\left(-\int_0^t \sigma(\mathbf{r}(s))ds\right)$ 。那么该点对像素的颜色贡献为 $T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt$. 通过对光线上每一点做积分, 我们得到该视点的像素颜色为:

$$\mathbf{C}(\mathbf{x}, \mathbf{d}) = \int_0^\infty T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt$$

在计算机中, 通过离散化积分, 假设采样点为 t_1, \dots, t_N , 则上式的离散结果为

$$\hat{\mathbf{C}}(\mathbf{x}, \mathbf{d}) = \sum_{i=1}^N T_i(1 - \exp(-\sigma_i \delta_i))\mathbf{c}_i$$

其中 $\delta_i = t_{i+1} - t_i$, $T_i = \exp\left(-\sum_{j=1}^{i-1} \sigma_j \delta_j\right)$

1.2 Neural Radiance Field

我们可以看到, 体渲染过程是可微分的。那么我们就可以通过神经网络拟合函数 $F : (\mathbf{x}, \mathbf{d}) \rightarrow (\mathbf{c}, \sigma)$, 使得我们可以通过位置和视角信息 (\mathbf{x}, \mathbf{d}) 得到视点的图像 $\hat{\mathbf{C}}(\mathbf{x}, \mathbf{d})$ 。Nerf

使用 MLP 实现这一过程。

对于输入的位置和视角信息 (\mathbf{x}, \mathbf{d}) , Nerf 使用 Positional Encoding 将其嵌入神经网络, 嵌入方式¹为:

$$\gamma(\mathbf{p}) = (\sin(2^0 \pi \mathbf{p}), \cos(2^0 \pi \mathbf{p}), \dots, \sin(2^{L-1} \pi \mathbf{p}), \cos(2^{L-1} \pi \mathbf{p}))$$

对于位置信息, 使用 $L = 10$; 对于方向信息, 使用 $L = 4$.

之后使用 MLP 将信息拟合为 (\mathbf{c}, σ) , 网络架构如图1. 每层全连接层后面都是 ReLU 激活函数。

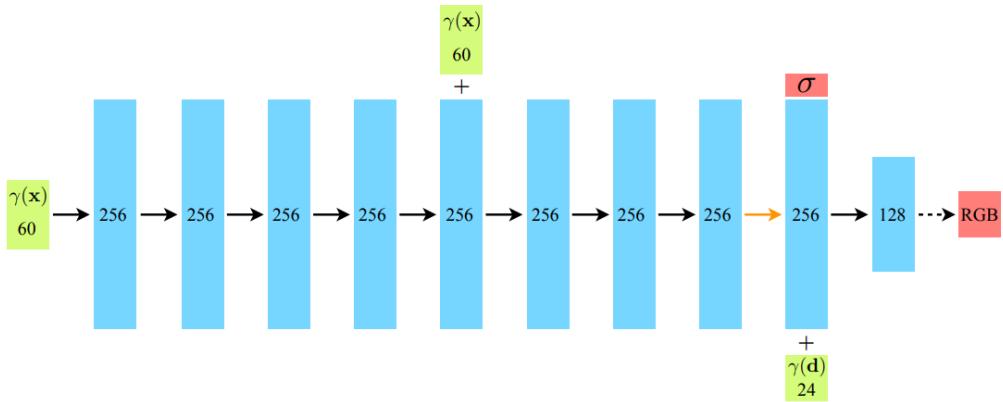


图 1: MLP network architecture

1.3 层次化采样 (Hierarchical volume sampling)

如果对光线进行等间隔采样, 那只能学到采样点上的信息, 所以在每个等长度区间里进行随机采样, 这一步称作粗采样。

在 Nerf 中, 粗 (coarse) 采样点设置为起点和终点之间等间隔的均匀采样:

$$t_i \sim U \left[t_n + \frac{i-1}{N} (t_f - t_n), t_n + \frac{i}{N} (t_f - t_n) \right]$$

然后将粗采样点使用 Coarse Network 做映射。

但是, 空间中有大量位置体密度几乎为 0, 因此如果只采用均匀采样会损失很多信息。对于 Coarse Network 的输出, 视点的像素颜色为

$$\hat{C}_{coarse}(\mathbf{r}) = \sum_{i=1}^{N_{coarse}} w_i \mathbf{c}_i$$

其中 $w_i = T_i(1 - \exp(-\sigma_i \delta_i))$

我们进一步根据权重 w_i 再进行一次细 (fine) 采样。首先对 w_i 做归一化得到概率分布 (piecewise-constant PDF) $\hat{w}_i = w_i / \sum_{j=1}^{N_{coarse}} w_j$ 。然后使用 inverse transform sampling 基于这些概率得到 N_{fine} 个细采样点, 并基于这些细采样点和粗采样点 (共 $N_{coarse} + N_{fine}$ 个) 优化 Fine Network。

层次化采样保证了我们能学到更多可视物体上的信息。

¹工程上实际上采用 $\gamma(\mathbf{p}) = (\sin(2^0 \pi \mathbf{p}), \cos(2^0 \pi \mathbf{p}), \dots, \sin(2^{L-1} \pi \mathbf{p}), \cos(2^{L-1} \pi \mathbf{p}))$

1.4 信息熵 (Ray Entropy)

Nerf 需要很多个视角才能渲染得到比较好的效果 [1], 我们希望通过比较少的视角得到还不错的结果。InfoNerf 考虑了光线的信息熵，因为只有少部分光线击中了视点，其余光线只起到噪音作用，所以 InfoNerf 最小化光线的信息熵。

$$H(\mathbf{r}) = - \sum_{i=1}^N p_i \log(p_i)$$

其中

$$p_i = \frac{1 - \exp(-\sigma_i \delta_i)}{\sum_{j=1}^N (1 - \exp(-\sigma_j \delta_j))}$$

，称为 Ray density。

对于没有击中任何物体的光线，信息熵为 0. 在优化过程中，我们丢弃到不经过任何物体的光线，定义为：

$$Q(\mathbf{r}) = \sum_{j=1}^N (1 - \exp(-\sigma_j \delta_j)) < \epsilon$$

由此，我们定义信息熵损失函数

$$L_{entropy} = \frac{1}{|R|} \sum_{\mathbf{r} \in R} \mathbf{1}\{Q(\mathbf{r}) > \epsilon\} H(\mathbf{r})$$

1.5 视角连续性

我们还希望维持视角的连续性，也就是说视角偏移一点我们希望出现的图像不要偏移太多。对于光线 \mathbf{r} ，进行略微偏移得到 $\tilde{\mathbf{r}}$ ，将其偏移程度用光线上各点 i 的 KL 衡量：

$$L_{KL} = \sum_{i=1}^N p(\mathbf{r}_i) \log \frac{p(\mathbf{r})}{p(\tilde{\mathbf{r}})}$$

这可以提高少样本学习的效果。

1.6 优化目标

基于上述讨论，我们给出 InfoNerf 的优化目标。

RGB Loss 显然，我们主要优化训练集的像素损失，即

$$L_{RGB} = \frac{1}{|R_s|} \sum_{\mathbf{r} \in R_s} \|C(\mathbf{r}) - \hat{C}(\mathbf{r})\|_2^2$$

其中 R_s 是训练集视点对应的光线。

Entropy Loss 基于 Nerf 的模型不能使用未见过的图像的光线，通过信息熵，我们能够优化未曾见过的视角。

$$L_{entropy} = \frac{1}{|R|} \sum_{\mathbf{r} \in R} \mathbf{1}\{Q(\mathbf{r}) > \epsilon\} H(\mathbf{r})$$

其中 $R = R_s \cup R_u$, R_s 是训练集视点对应的光线， R_u 是随机采样的不在测试集中的光线。

Information Gain Reduction Loss 对相机姿态做 $\pm 5^\circ$ 的扰动，并计算其损失。

$$L_{KL} = \sum_{i=1}^N p(\mathbf{r}_i) \log \frac{p(\mathbf{r})}{p(\tilde{\mathbf{r}})}$$

对于，InfoNerf 优化 $L_{RGB} + \lambda_1 L_{entropy} + \lambda_2 L_{KL}$.

2 训练环境和参数设置

用 Jittor 实现，在单张 NVIDIA 1080Ti(12GB) 上训练了 50000 iter，测试集上得到

$$PSNR_{old} = 19.126, PSNR_{new} = 19.315, MSE = 0.012$$

其中 $PSNR_{old}$ 是 Nerf 的计算方式，它通过平均 MSE 计算 PSNR； $PSNR_{new}$ 是本文的计算方式，通过每个图片对的 MSE 计算 PSNR，最后取平均。复现结果和论文中的结果一致。

我们使用文章 [1] 中预设的参数进行训练：初始学习率 $5e-4$ ；优化器为 Adam($\beta = (0.9, 0.999)$)；

每个光线上的粗采样点为 64，细采样点为 128；Seen rays 和 Unseen rays(for entropy loss) 的数量都设置为 1024，认为不可见的阈值 $\epsilon = 0.1$ 。 $\lambda_1 = 0.001, \lambda_2 = 0^2$ 。此外，对于 Entropy loss，我们实际上只优化了 Unseen rays³。

训练方式采用少样本学习，训练集只有 4 个角度的图像，编号为 [8, 72, 37, 41]。验证集共 40 张。测试集中每隔 8 张取一张，共 25 张。

3 训练结果与分析

训练 50000 步后，得到不同训练步数、不同视角下的测试集图像如图2。可以看到，我们的结果达到了原论文声称的效果，lego 模型上比较精细的凸起也能够构建出来。此外，模型的正面效果比背面效果要好，这可能是因为正面的细节比较多，神经辐射场更容易拟合细节较少的视角，也可能是因为训练集缺乏方面视角的样本，毕竟只有 4 张。模型侧后方的体渲染出现了白背景上的噪点，这可能是训练样本没有覆盖该视角。总体来说，算法通过引入 Regularization by Ray Entropy Minimization and Information Gain Reduction，在少样本的 lego 模型上超过了原有的 nerf。

mp4 格式的视频见作业附件 `lego_spiral_050000_rgb.mp4`。

参考文献

- [1] Mijeong Kim, Seonguk Seo, and Bohyung Han. Infonerf: Ray entropy minimization for few-shot neural volume rendering. *CoRR*, abs/2112.15399, 2021.

²对于为何不使用 L_{KL} ，作者解释到“For the synthetic dataset, the information gain reduction loss is not necessary because the sampled images usually have **wide-baseline**。”见<https://github.com/mjmjeong/InfoNeRF/issues/2>。我们的训练集的 4 张照片便是 wide-baseline。

³对于为何不优化 Seen Rays，作者解释到“In the few-shot setting, since the distribution of R_u itself is almost close to the whole ray distribution, using only R_u is enough. Also, R_s is sampled from the few selected views while R_u is sampled from almost full views. Then, probabilistically, one ray from seen views is sampled more frequently than another ray from unseen views. In this regard, we think the potential unbalance optimization issue can happen, so we modified it for better stabilization.”，见<https://github.com/mjmjeong/InfoNeRF/issues/3>

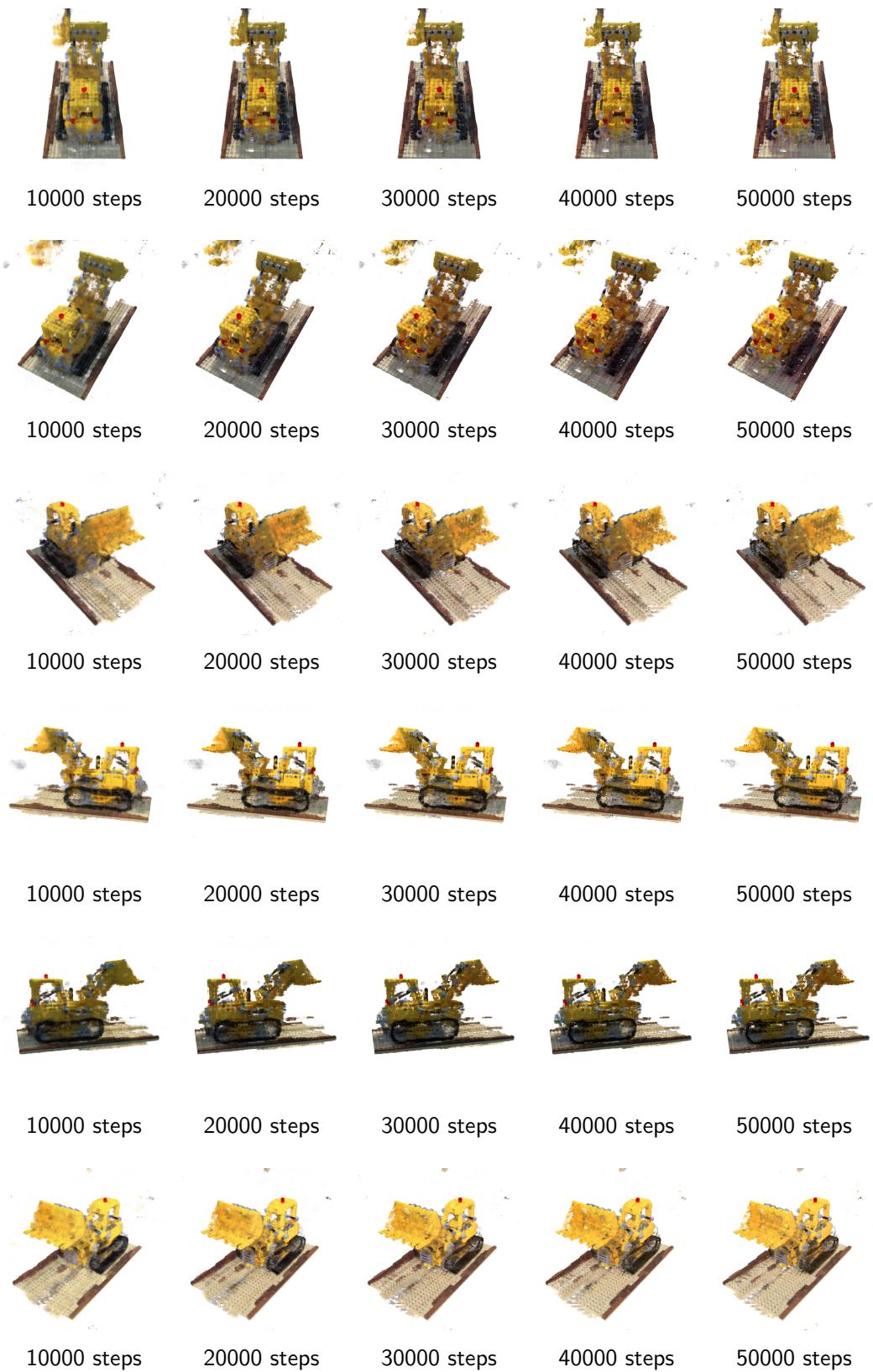


图 2: Lego 的算法输出

- [2] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *CoRR*, abs/2003.08934, 2020.