NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis

- https://arxiv.org/abs/2003.08934
- https://github.com/bmild/nerf
- [v1] Thu, 19 Mar 2020 17:57:23 UTC (7,448 KB)

1 Introduction

5D input: 3D coordinates (x, y, z) + 2D viewing direction (θ, ϕ)

To render *neural radiance field*:

1. march camera rays through the scene to generate a sampled set of 3D points
2. use those points and their corresponding 2D viewing directions as input to the neural network to produce an output set of colors and densities
3. use classical volume rendering techniques to accumulate those colors and densities into a 2D image

Loss: error between each observed image and the corresponding views rendered from our representation

Positional encoding:

- deep networks 更倾向于学习低频的函数，把输入"加工"成高频的可以解决这个问题
  - 超参数L决定了神经网络能学习到的最高频率的大小
- 基本的concatenation不能在高分辨率的情况下收敛，效率低
- 多级可导便于插值等操作这个性质，应该是在体素表示等方法里面比较有用，在这里无关

> 把整个物体建模成了一个函数，通过输入得到density和RGB

3 Neural Radiance Field Scene Representation

- 5D input to 6D input
  - 3D coordinates (x, y, z) + 2D viewing direction (θ, ϕ)
  - 3D coordinates is denoted by $\mathbf{x}$
  - 2D viewing direction is expressed as a 3D Cartesian unit vector $\mathbf{d}$
- 4D output
  - 3D emitted color (r, g, b) + volume density
  - 3D emitted color is denoted by $\mathbf{c}$
  - volume density is denoted by σ
- MLP network: $F_\Theta : (\mathbf{x}, \mathbf{d}) \rightarrow (\mathbf{c}, \sigma)$
  - Θ is optimized

    - to guarantee multi-view consistency
        - σ is a function of $\mathbf{x}$, denoted by $\sigma(\mathbf{x})$
        - $\mathbf{c}$ is a function of $\mathbf{x}$ and $\mathbf{d}$, denoted by $\mathbf{c}(\mathbf{x}, \mathbf{d})$

> Such a representation can be used to storing an model
>
> Details of the architecture will be demonstrated later.

4 Volume Rendering with Radiance Fields

Theory:

- The volume density $\sigma(\mathbf{x})$ can be interpreted as the differential probability of a ray **terminating** at an infinitesimal particle at location $\mathbf{x}$

> 光线终止在这个点的概率，且这个概率是可以微分计算的

- camera ray $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$
    - $\mathbf{o}$: 相机的光心
    - $\mathbf{d}$: viewing direction
    - t: 点与光心的距离
    - 约等于表示以另一个坐标系表示点的坐标
- accumulated **transmittance** along the ray from **near** to current point $T(t) = \exp\left(-\int_{t_n}^{t} \sigma(\mathbf{r}(s))ds\right)$
    - t_n: **near** distance
    - t_f: **far** distance
    - that is, the probability that the ray travels from t_n to t without hitting any other particle
    - $T(t)$代表的则是光线在走到$\mathbf{x}$时，没有碰撞到任何粒子的概率，换一个角度说也就是有占比$T(t)$的光还可以用来在当前点产生颜色（实际在当前点用了多少由$T(t)\sigma(\mathbf{r}(t))$）。其中点$\mathbf{x}$用点到光心的距离$t$表示。

> NeRF的模型假设是，光线与粒子碰撞而停下，并且发生让人感知到颜色的吸收、反射等行为

- expected color of a ray $C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt$
    - $T(t)\sigma(\mathbf{r}(t))$表示了有多少之前未碰撞的光在当前点发生碰撞

Practice:

- computer cannot calculate such continuous integration
- use hierarchical volume sampling: coarse sampling + fine sampling
- coarse sampling for expected color $\hat{C}(\mathbf{r}) = \sum_{i=1}^{N} T_i(1 - \exp(-\sigma_i\delta_i))\mathbf{c}_i$
    - sample N points in an evenly-spaced fashion between t_n and t_f
    - distance between adjacent samples $\delta_i = t_{i+1} - t_i$
    - **transmittance** $T_i = \exp(-\sum_{j=1}^{i-1} \sigma_j\delta_j)$
    - trivially differentiable

- **reduce the calculation** to traditional alpha composition with alpha $\alpha_i = 1 - \exp(-\sigma_i \delta_i)$

5 Optimizing a Neural Radiance Field

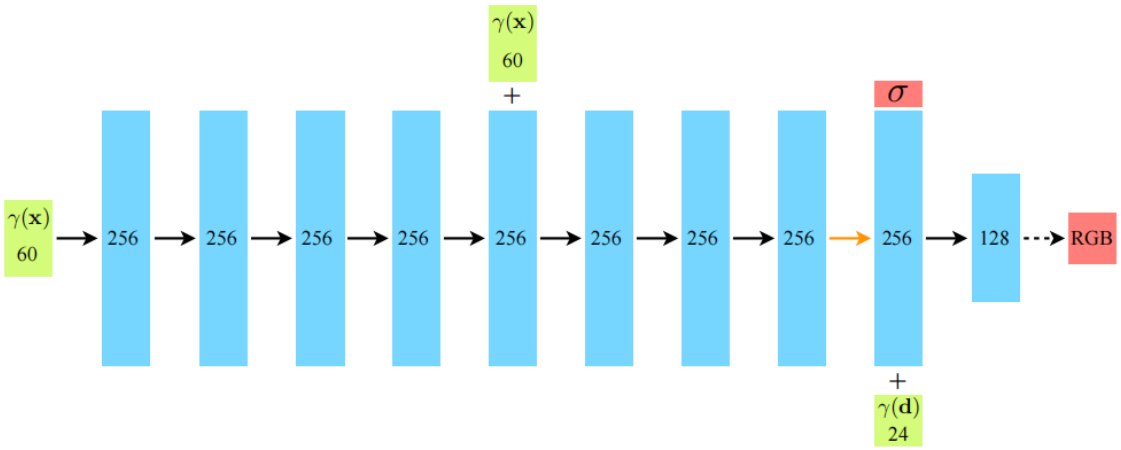
5.1 Positional encoding

- deep networks 更倾向于学习低频的函数，把输入“加工”成高频的可以解决这个问题
  - 超参数L决定了神经网络能学习到的最高频率的大小
- 基本的concatenation不能在高分辨率的情况下收敛，效率低
- reformulating the network F_Θ as a composition of two functions $F_\Theta = F'_\Theta \circ \gamma$
  - F'_Θ , a regular MLP, is learned
  - γ is positional encoding $\mathbb{R} \to \mathbb{R}^3$
    - $\gamma(p) = (\sin(2^0 \pi p),\ \cos(2^0 \pi p), \ldots,\ \sin(2^{L-1} \pi p),\ \cos(2^{L-1} \pi p))$
    - applied separately to each of the three coordinate values in $\mathbf{x}$ (which are normalized to lie in [−1, 1])
    - applied separately to the three components of the Cartesian viewing direction unit vector $\mathbf{d}$(which by construction lie in [−1, 1])
    - L is a hyper-parameter and is set $L = 10$ for $\gamma(\mathbf{x})$ and $L = 4$ for $\gamma(\mathbf{d})$
    - a space coordinate becomes **60D** and a viewing direction becomes **24D**
- we use these functions to map continuous input coordinates into a higher dimensional space to enable our MLP to more easily approximate a higher frequency function

5.2 Hierarchical volume sampling

- Instead of just using a single network to represent the scene, we simultaneously optimize two networks: one “coarse” and one “fine”
  - 源码里根据参数`N_importance`决定是否定义`network_fine`，两个网络的参数一起放入`grad_vars`，优化器直接优化`grad_vars`
- in coarse network, sample **uniformly**
  - sample N_c points in an **evenly-spaced fashion** between t_n and t_f
  - rewrite alpha composited color **from coarse network** as a **weighted form** $\hat{C}_c(\mathbf{r}) = \sum_{i=1}^{N_c} w_i c_i$ with $w_i = T_i(1 - \exp(-\sigma_i \delta_i))$
  - the result with normalization $\hat{w}_i = \frac{w_i}{\sum_{j=1}^{N_c} w_j}$ forms a new distribution along the ray, which is a probability density function (PDF)
- in fine network, sample according to the distribution from coarse network
  - use **inverse transform sampling** to sample N_f points from the distribution from coarse network
  - use $N_c + N_f$ points to compute the final result $\hat{C}_f(\mathbf{r})$

5.3 Implementation details



> Input vectors are shown in green, intermediate hidden layers are shown in blue, output vectors are shown in red, and the number inside each block signifies the vector's dimension.
>
> All layers are **standard fully-connected layers**, *black arrows* indicate layers with ReLU activations, *orange arrows* indicate layers with no activation, *dashed black arrows* indicate layers with sigmoid activation, and "+" denotes vector concatenation. The positional encoding of the input location $\gamma(\mathbf{x})$ is passed through 8 fully-connected ReLU layers, each with 256 channels. We follow the DeepSDF architecture and include a skip connection that concatenates this input to the fifth layer's activation.
>
> An additional layer outputs the volume density σ (which is rectified using a ReLU to ensure that the output volume density is nonnegative) and a 256-dimensional feature vector. This feature vector is concatenated with the positional encoding of the input viewing direction $\gamma(\mathbf{d})$ and is processed by an additional fully-connected ReLU layer with 128 channels.
>
> A final layer (with a sigmoid activation) outputs the emitted RGB radiance at position $\mathbf{x}$, as viewed by a ray with direction $\mathbf{d}$

- We use ground truth camera poses, intrinsics, and bounds for synthetic data, and use the COLMAP structure-from-motion package to estimate these parameters for real data
- At each optimization iteration
  - we **randomly sample a batch of camera rays** from the set of all pixels in the dataset, and then follow the hierarchical sampling described to query N_c samples from the coarse network and $N_c + N_f$ samples from the fine network
  - We then use the volume rendering procedure described to render the color of each ray from both sets of samples
  - Loss is the MSE for the two networks:
    $\mathcal{L} = \sum_{\mathbf{r}\in\mathcal{R}} \left[\|\hat{C}_c(\mathbf{r}) - C(\mathbf{r})\|_2^2 + \|\hat{C}_f(\mathbf{r}) - C(\mathbf{r})\|_2^2\right]$
    - $\mathcal{R}$ is the set of rays in each batch
    - the loss of coarse network is also optimized so that the weight distribution can be used to allocate samples in the fine network
- a space coordinate becomes **60D** and a viewing direction becomes **24D** after positional encoding
- **Volume Bounds**
  - For experiments with synthetic images, we scale the scene so that it lies within **a cube of side length 2 centered at the origin**, and only query the representation within this bounding volume

    - For real images, we use normalized device coordinates to map the depth range of these points into [−1, 1]
    - This shifts all the ray origins to the near plane of the scene, maps the perspective rays of the camera to parallel rays in the transformed volume, and uses disparity (inverse depth) instead of metric depth, so all coordinates are now bounded

> Refer to the *NDC ray space derivation* of the original paper for more about normalized device coordinates
>
> 主要是讲了空间坐标变换对光线$\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$的影响

- **Training Details**
    - For real scene data, we regularize our network by adding random Gaussian noise with zero mean and unit variance to the output σ values before passing them through the ReLU
    - slightly improves visual performance for rendering novel views

6 Results

...

7 Conclusion

- future directions
    - investigate techniques to efficiently optimize and render neural radiance fields
    - interpretability: sampled representations such as voxel grids and meshes admit reasoning about the expected quality of rendered views and failure modes

Possible Direction

1. editability — explicit control
2. speed — faster optimization and render
3. generalization
4. scalability
5. dynamic — used in video
6. expensive training — how to use less or cheaper training samples
7. better performance — less artifacts and more accurate color
8. application

Other Resources

1. NeRF及其发展
2. Neural Radiance Fields (NeRF)
3. Volume Rendering 公式推导
4. NeRF 论文主要点细致介绍

5. Coordinate Systems
6. 从齐次裁剪空间到NDC
7. 【NeRF】背景、改进、应用与发展_Summer tree的博客-CSDN博客