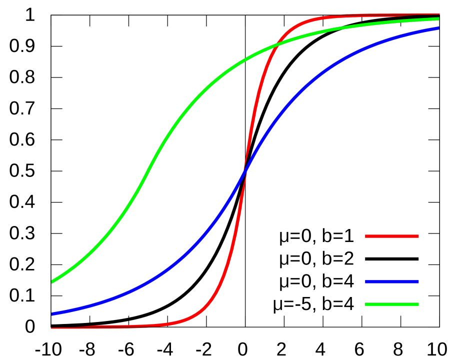
Paper Summary Volume Rendering of Neural Implicit Surfaces

Abstract

Improve geometry representation and reconstruction in neural volume rendering. We achieve that by **modeling the volume density as a function of geometry**. This is in contrast to previous work modeling the geometry as a function of the volume density.

Define the volume density function as **Laplace's CDF applied to a signed distance function** representation.



$$\text{Laplace CDF} = \begin{cases} \frac{1}{2}exp(\frac{x-\mu}{b}) & \text{if } x \leq \mu \\ 1 - \frac{1}{2}exp(\frac{x-\mu}{b}) & \text{if } x \geq \mu \end{cases}$$

where we should treat the value of signed distance function as x in this CDF. Three benefits for this representation:

1. Providing a useful inductive bias to the geometry learned in the neural volume rendering process.

2. Facilitating a bound on the opacity approximation error, leading to an accurate sampling of the viewing ray.

3. Allowing efficient unsupervised disentanglement of shape and appearance in volume rendering.

Understanding for disentanglement:

1. Introduction

Recall the volume rendering integral:

$$C(r) = \int_{t_n}^{t_f} T(t)\sigma(r(t))c(r(t), d)dt$$

Motivation: the density part $\sigma(\mathbf{x})$ is not as successful in faithfully predicting the scene's actual geometry, often producing noisy, low fidelity geometry approximation.

We propose VolSDF. The **key idea** is representing the density $\sigma(x)$ as a function of the signed distance to the scene's surface.

Benefits:

1. Guarantees the existence of a well-defined surface that generates the density.
2. Bound the approximation error of the opacity along rays when sampling the viewing ray.

2. Related Work

a. Neural Scene Representation & Rendering: It is non-trivial to find a proper threshold to **extract surfaces** from the predicted density.

3. Method

a. Density as transformed SDF

$$\mathbf{1}_{\Omega}(\mathbf{x}) = \begin{cases} 1 & \text{if } x \in \Omega \\ 0 & \text{if } x \textit{ not} \in \Omega \end{cases}$$

and $d_{\Omega}(\mathbf{x}) = (-1)^{1_{\Omega}(x)} \min_{y \in M} ||\mathbf{x} - \mathbf{y}||$

$$\sigma(x) = \alpha \Psi_{\beta}(-d_{\Omega}(x))$$

where

$$\Psi_{\beta}(s) = \begin{cases} \frac{1}{2} exp(\frac{s}{\beta}) & \text{if } s \leq 0 \\ 1 - \frac{1}{2} exp(\frac{-s}{\beta}) & \text{if } x > 0 \end{cases}$$

$\alpha, \beta > 0$ are learnable parameters, and Ψ_{β} is the CDF of Laplace distribution with zero mean and β scale. A constant density α smoothly decreases near the boundary, where the smoothing amount is controlled by β. The benefit of defining the density in this way is:

1. Providing a principled way to reconstruct the surface, i.e., as the zero level set of d_{Ω}.
2. Bounding error.

3.2 Volume Rendering of σ

Volume rendering is all about approximating the integrated (**summed**) light radiance along this ray reaching the camera. Two important quantity in computation: **volume's *opacity*** O, or equivalently, its *transperancy* T, and **radiance field** L, (or $c(r(t), d)$ in NeRF).

$$T(t) = exp(-\int_0^t \sigma(\mathbf{x}(s)) ds)$$

and the *opacity* O is the complement probability

$$O(t) = 1 - T(t)$$

Note that O is a monotonic increasing function where $O(0) = 0$ and assuming that every ray is eventually occulded $O(\infty) = 1$. Then we can think of O as a CDF, and

$$\tau(t) = \frac{dO}{dt}(t) = \sigma(\mathbf{x}(t))T(t)$$

is its PDF. The volume rendering equation is the expected light along the ray, **Used in this Paper**

$$I(\mathbf{c}, \mathbf{v}) = \int_0^\infty L(\mathbf{x}(t), \mathbf{n}(t), \mathbf{v})\tau(t)dt$$

where L is the radiance field. Comparing what I know from NeRF:

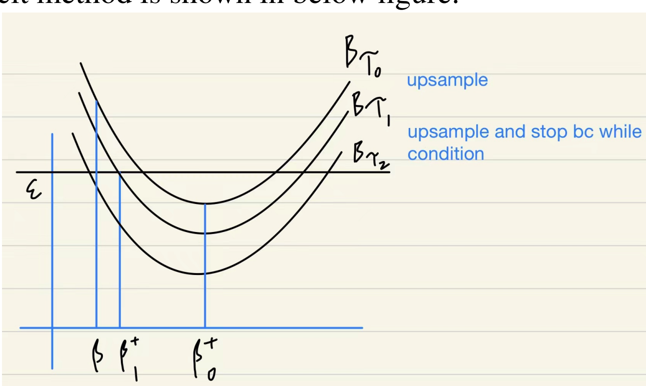
$$C(r) = \int_{t_n}^{t_f} T(t)\sigma(r(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt$$

where $\mathbf{c}(\mathbf{r}(t), \mathbf{d})$ is the radiance field

$I(\mathbf{c}, \mathbf{v})$ and $\mathbf{C}(\mathbf{r})$ are actually the same expressions, just using different name.

3.4 Sampling Algorithm

In a nutshell, we start with a uniform sample $T = T_0$, and use Lemma 2 to initially set a $\beta_+ > \beta$ that satisfies $B_{T,\beta_+} \leq \varepsilon$. Then, we repeatedly upsample T to reduce β_+ while maintaining $B_{T,\beta_+} \leq \varepsilon$. The explicit method is shown in below figure.



3.5 Training

$*$ Also positional encoding for the position $\mathbf{x}$ and view direction $\mathbf{v}$ in the geometry and radiance field.

System consists of two MLP:

1. f_φ approximating the SDF of the learned geometry, as well as global geometry feature z of dim 256: $f_\varphi(x) = (d(x), z(x)) \in R^{1+256}$.
2. $L_\psi \in R^3$ representing the scene's radiance field with learnable parameters ψ, where R^3 is (r, g, b).

Next, how f_φ is training.

Overall, The performance of their proposed sampling algorithm is really good, but only work for the density function they use. Coarse and fine sampling is more general, but like they mentioned in paper, sampling points derived from another network cannot guarantee correct approximation. The highlight for this paper is we can extract object by setting the value of SDF = 0.