DualSDF

Thursday, January 7, 2021 7:44 PM

Intro

- We propose a generative two-level model that simultaneously represents 3D shapes using two levels of granularity, one for capturing fine-grained detail, the other for encoding a coarse structural decomposition.
- The two levels are tightly coupled via a shared latent space, wherein a single latent code vector decodes to two representations of the same shape. Modifications to one representation can be propagated to the other via the shared code.
- The shared latent space is learnt with a variational auto decoder (VAD). This approach
  1. imposes a Gaussian prior on the latent space, which enables sampling,
  2. encourages a compact latent space suitable for interpolation and optimization based manipulation.

Coarse Primitive-based shape representation.

- A SDF specifies, for every point $p = (p_x, p_y, p_z)$, the distance from that point to the nearest surface, where the sign encodes whether the point is inside or outside.
- Denote a set of N basic shape primitives by tuples:

$$\{(c^i, \alpha^i) \mid i = 1 \dots N\}$$

where c^i describes the primitive type, and $\alpha^i \in \mathbb{R}^{k^i}$ describes the attributes of the primitives.
The dimentionality k^i denotes the dof for primitive i.
The SDF of a single element i is

$$d_{c^i}(p, \alpha^i) = SDF_{c^i}(p, \alpha^i)$$

An example of a simple geometric primitive is sphere;

$$k^{sphere} = 4$$

$$\alpha^{sphere} = [c, r]$$

$c = (c_x, c_y, c_z)$ is center

r is radius

$$d_{sphere}(p, \alpha^{sphere}) = \|p - c\|_2 - r$$

- to approximate the SDF of an arbitrarily complex shape, we construct the SDF of the union of geometric elements (spheres)

$$\alpha = [\alpha^1, \dots, \alpha^N]$$

$$d_C(p, \alpha) = \min_{1 \le i \le N} d_{c^i}(p, \alpha^i)$$

- To train the primitive-based model, given a target space x, eg. mesh, sample pairs of 3D points p_t and their gt SDF $s_t = SDF_x(p_t)$, α can be learnt by minimizing the difference between predicted and real SDF:

$$\hat{\alpha} = \arg\min_{\alpha} \sum_t L_{SDF}(d_C(p_t, \alpha), s_t).$$

High resolution shape representation.

Same with DeepSDF. directly learn the SDF with a neural network g_ϕ:

$$g_\phi(p) \approx SDF_x(p).$$

Learning a tightly coupled latent space.

- We learn a two-level shape representation over an entire class of shapes $\{x_j \mid j = 1 \dots M\}$ by using two representation models that share the same latet code z_j.

- For representing multiple shapes with the primitive based coarse-level representation, we parameterize α with a neural network f_θ:

$$\alpha_j = f_\theta(z_j)$$

f_θ is shared for all shapes.

For the fine-scale, we condition the neural network g_ϕ on latent code z_j:

$$g_\phi(z_j, p) \approx SDF_{x_j}(p).$$

VAD enforces a strong regularization on the latent space by representing the latent vector of each individual shape z_j with the parameters of its approximate posterior distribution (μ_j, σ_j). $\Downarrow$

- We select the family of Gaussian distributions with diagonal covariance matrix as the approximate posterior of z, given shape x_j:

$$q(z \mid x = x_j) = \mathcal{N}(z; \mu_j, \sigma_j^2 \cdot I)$$

We apply the reparameterization trick, sampling $\epsilon \sim \mathcal{N}(0, I)$ and setting $z_j = \mu_j + \sigma_j \odot \epsilon$ to allow optimization of μ_j, σ_j via GD.
During training, we max. the lower bound of the marginal likelihood (ELBO) over the dataset, which is the sum over lower bound of each individual shape x:

$$\log p_{\theta,\phi}(x) \ge E_{z \sim q(z|x)}[\log p_{\theta,\phi}(x \mid z)] - D_{KL}(q(z|x) \,\|\, p(z)).$$

The learnable parameters are θ, ϕ, and variational parameters $\{(\mu_j, \sigma_j) \mid j = 1 \dots M\}$, that parameterize $q(z|x)$.

- Since we'd like the two representations to be tightly coupled, i.e. both assign high probability density to a shape x_j given its latent code $z_j \sim q(z \mid x = x_j)$, we use a mixture model:

$$p_{\theta,\phi}(x \mid z) = \frac{p_\theta(x|z) + p_\phi(x|z)}{2}$$

$p_\theta(x|z)$, $p_\phi(x|z)$ are the posterior distributions of coarse and fine representations.

$$\left.\begin{aligned} \log p_\theta(x|z) &= -\lambda_1 \int p(p)\, L_{SDF}(d_C(p, f_\theta(z)), SDF_x(p))\, dp \\ \log p_\phi(x|z) &= -\lambda_2 \int p(p)\, L_{SDF}(g_\phi(z, p), SDF_x(p))\, dp. \end{aligned}\right\}$$

Eq. above can be approximated via Monte Carlo, where p is sampled randomly from the 3D space following a specific rule $p(p)$.