Paper Summary Deep SDF

Abstract

Deep SDF enables high quality shape reconstruction, interpolation, and completion from **partial** and **noisy** 3D input data. Implicitly encodes a shape's boundary as the zero-level-set of the learned function.

1. Introduction

Fully continuous. Generative. Learned shape-conditioned classifier for which the decision boundary is the surface of the shape itself. Contribution:

1. The formulation of generative shape-conditioned 3D modeling with a continuous implicit surface.
2. $*$ A learning method for 3D shapes based on a probabilistic auto-decoder.
3. Shape modeling and completion.

2.2 Representation Learning Techniques

Optimizing Latent Vectors. An alternative is to learn compact data representations by training *decoder-only* networks. Simultaneously optimizes the latent vectors assigned to each point and the decoder weights through back-propagation. Auto-decoders are trained with *self-reconstruction* loss on decoder-only architecture.

3. Modeling SDFs with Neural Networks

Describe modeling shapes as the zero iso-surface decision boundaries of feed-forward networks trained to represent SDFs. Continuous function that, for a given spatial point, outputs the point's distance to the closet surface, whose sign encodes whether the point is inside or outside of the watertight surface:

$$SDF(x) = s : \mathbf{x} \in R^3, s \in R$$

Key idea is to directly regress the continuous SDF from point samples using DNN. Such surface representation can be intuitively understood as a learned **binary classifier** for which the decision boundary is the surface of the shape itself. Given a target shape, we prepare a set of pairs X composed of 3D point samples and their SDF values:

$$X := \{(\mathbf{x}, s) : SDF(\mathbf{x}) = s\}$$

$$f_\theta(\mathbf{x}) \approx SDF(\mathbf{x}),\ \forall \mathbf{x} \in \Omega$$

where Ω is target domain. Training is done by minimizing the sum over losses b/t the predicted and real SDF values of points in X under the following L_1 loss function:

$$L(f_\theta(\mathbf{x}), s) = |clamp(f_\theta(\mathbf{x}), \delta) - clamp(s, \delta)|$$

where δ control the distance from the surface over which we expect to maintain a metric SDF.
larger values of δ allow for fast ray-tracing since each sample gives information to safe step sizes.
smaller values of δ can be used to concentrate network capacity on details near the surface.

Accurate normals can be analytically computed by calculating the spatial derivative $\frac{\partial f_\theta(\mathbf{x})}{\partial \mathbf{x}}$ via back-propogation through the network.

4.2 Auto-decoder-based DeepSDF Formulation