DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation

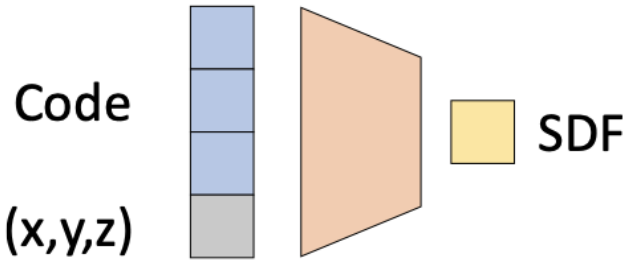
Report

Introduction

Most of the previous works were based on mainly 3 types of 3D representations: Voxels, Mesh and Point-Based. Point-Based Representation is a sparse representation of the 3d object. They lack continuity. Voxel-based reconstruction describes volumes with 3d grids of values. They consume a lot of memory and hence we limit them to 128x128x128 resolutions. Mesh representation are not continuous and are not closed

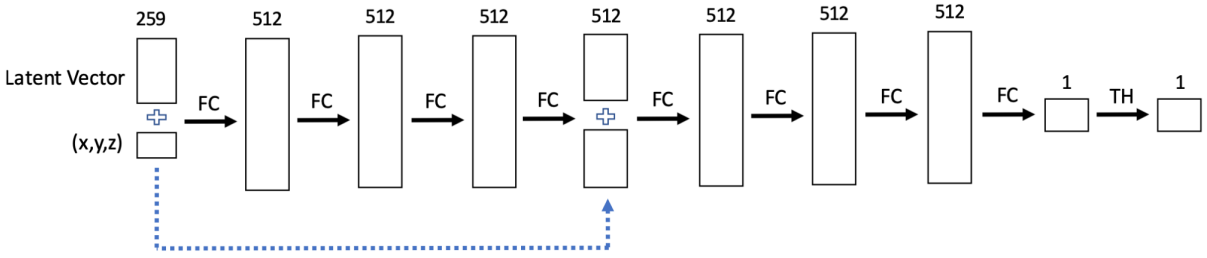
These representations are good but these are not visually compelling. The authors introduce DeepSDF, an implicit representation, where using Deep neural networks, we find a function f which is close to the zero iso-surface of the 3d model. The function takes in 3d coordinate as input and outputs the nearest distance of the point from the surface of the 3D object. The points which lie within are represented as negative distance and points outside are represented as positive distance

DeepSDF



Our goal is to predict the SDF of a point coordinate using deep neural networks. After predicting the SDF, we can find the zero iso-surface by sampling spatial points. It is basically a binary classifier with the decision boundary giving the surface of the 3D object. SDF gives more information than occupancy network because SDF also has additional information about the distance of each point from the surface

The SDF is made up of 8 fully connected layers of 512 dimension with ReLU activation as the non-linearity of function. For the final layer, we use the Tanh activation function which represents the SDF of a point coordinate input. In the experiments, batch normalization proved to be unstable, so they use weight normalization. Once trained, the surface is implicitly represented as the zero iso-surface of fθ(x), which can be visualized through raycasting or marching cubes.



Loss Function

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

$$\operatorname{clamp}(x, \delta) := \min(\delta, \max(-\delta, x))$$

We use the L1 loss function, and we clamp the distance between -δ and δ. We use $\delta = 0.1$. So we don't consider points whose distance is more than 0.1 units from the surface of the object.

End Note

This paper was the first paper to introduce SDF in 3d modelling using a Deep Neural Network.

To know more about DeepSDF, check out the paper: DeepSDF