ATIV tutorial - Implicit neural reconstruction

Julie Digne

November 28th 2024

In this lab work we will reconstruct shapes from point sets with and without their normal information.

Each network will output the distance or signed distance and one can extract the surface with Marching cubes, following these steps :

— Use the trained network to compute the values of the signed distance on a grid
— Extract the 0 levelset (marching_cubes method of the mcubes library)
— Save and visualize the mesh (export_obj method of the mcubes library)

1 A traditional reconstruction approach [Hoppe 92], as a baseline

This method is a "historical" method for reconstructing a surface from a set of points. It consists in taking an oriented point cloud $(x_i, \mathbf{n}_i)$, and estimating for any arbitrary point x in the ambient space a signed distance function as : $u(x) = \pm \min_i \|x_i - x\|$

The sign is given by the sign of the scalar product $\langle x - x_i, n_i \rangle$.

The original method starts with unoriented point clouds and devises a clever way to estimate the normal direction and their orientation. Here, for simplicity, we start with oriented points.

In this first part, we'll follow the method to build a baseline for surface reconstruction :

1. Load the oriented point set
2. Build a kdtree encoding this point set(spatial.KDTree in scipy.spatial)
3. Build a grid of points (np.mgrid)
4. Compute the signed distance values for each point of this grid using the kdtree.
5. Extract the 0 level set (marching_cubes of the mcubes library)
6. Save and visualize the mesh (export_obj of the mcubes library)

2 DeepSDF [Park 2019]

2.1 MLP Architecture

— MLP with 8 layers, $d_1 = 3, d_2 = 512, d_3 = 512, d_4 = 509, d_5 = 512, d_6 = 512, d_7 = 512, d_8 = 1$
— Skip connection from the first layer to the fourth layer.
— Activation functions : ReLU except for the last one $\varphi(a) = tanh(a)$

2.2 Loss

The loss is computed by sampling points in the ambient space (set X), computing their ground truth SDF ($gtsdf$) using a KdTree (as in part one), and fitting the loss :

$$\mathcal{L}(\theta) = \mathbb{E}_{x \sim X}[clamp_\delta(u_\theta(x)| - clamp_\delta(gtsdf(x)))]$$

σ is a parameter that you can play with.

3 Surface reconstruction from signed distance with Eikonal regularization [Gropp 2020]

3.1 Data

A set of points x_i with normals $\mathbf{n}_i$ sampled on the object surface.

3.2 Architecture

— MLP with 6 layers and 128 neurons per layer
— ReLU activation except for the last layer (linear)

3.3 Loss

— Data attachment loss

$$\sum_i \|u_\theta(x_i)\|^2 + (1 - \langle \mathbf{n}_i, \nabla u_\theta(x_i) \rangle)^2$$

— Eikonal loss

$$\mathbb{E}_x[|1 - \|\nabla u_\theta(x)\|_2^2|]$$

This expectation is approximated using a Monte Carlo estimation, at each epoch points are sampled in the ambient space and we evaluate the sum at these points.