Robust Neural Fields - NN Distance Extractors

Julian Strietzel, Ata Jadid Ahari under the supervision of Ziya Erkoc

Visual Computing Group
Prof. Matthias Nießner

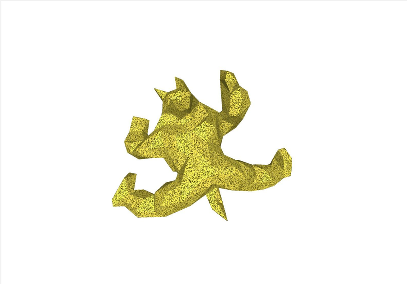
Problem Statement

Neural implicit fields are the Sota 3d object representation methods

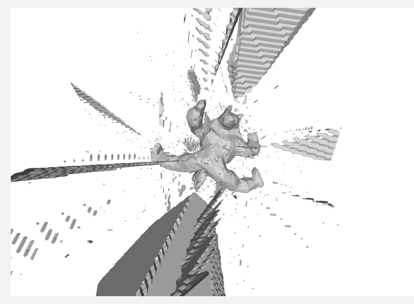
➩ Represent the objects in the weights of a NN

Recent methods rely on Normals to overfit the NN ➩ Shapes with bad normals cannot be represented robustly

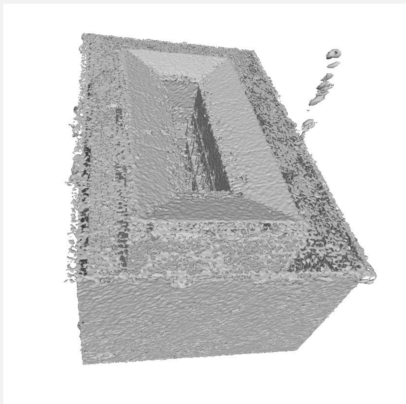
Inverted normals (5%)



SDF overfitting

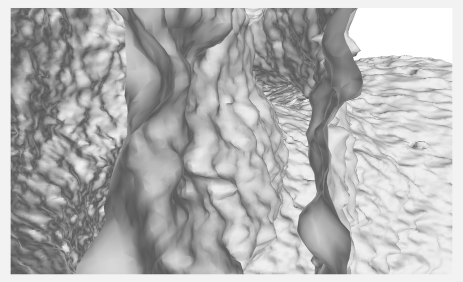


Inherently bad normals on edges



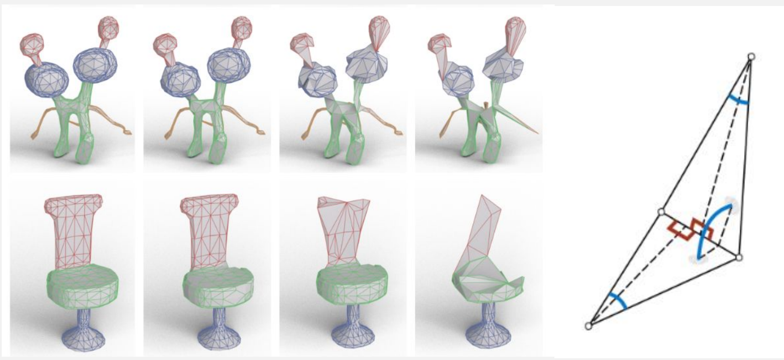
Unsigned distance field as a potential solution → calculate the distance mathematically without the sign

→ hard to reconstruct using M.-cubes



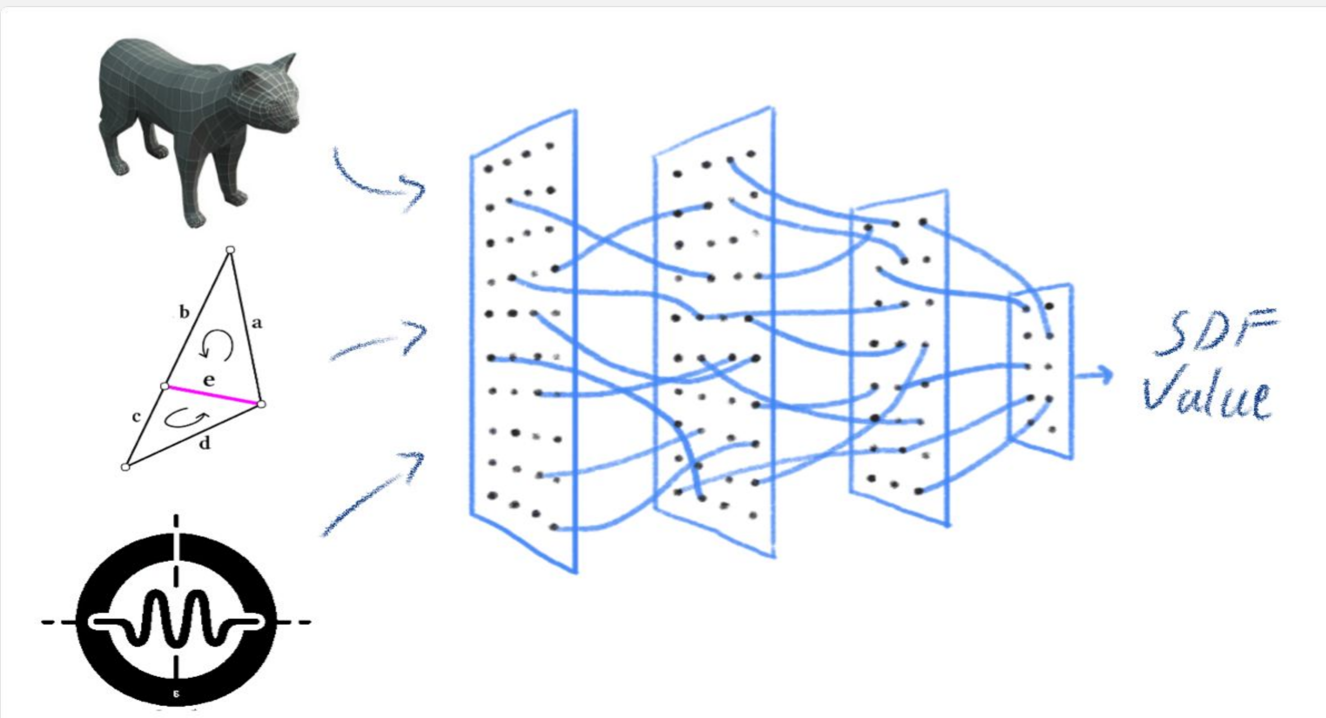
MeshCNN

➩ Generalizing Convolutional Neural Networks to 3D-Meshes



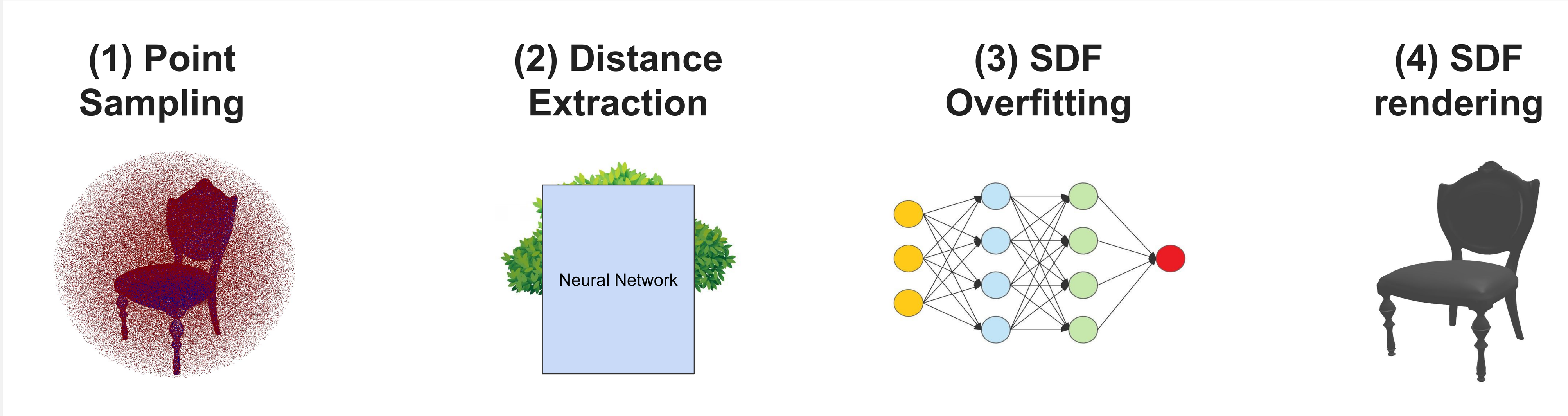
We use MeshCNN and other NN architectures as **Distance Extractors** in an SDF overfitting pipeline, to exchange standard methods with the explained shortcomings.

As input the get information about the Mesh and a positional encoding, to extract the distance. We are training those on simple shapes and expect them to generalize to badly defined shapes, as they are not relying on normal quality.



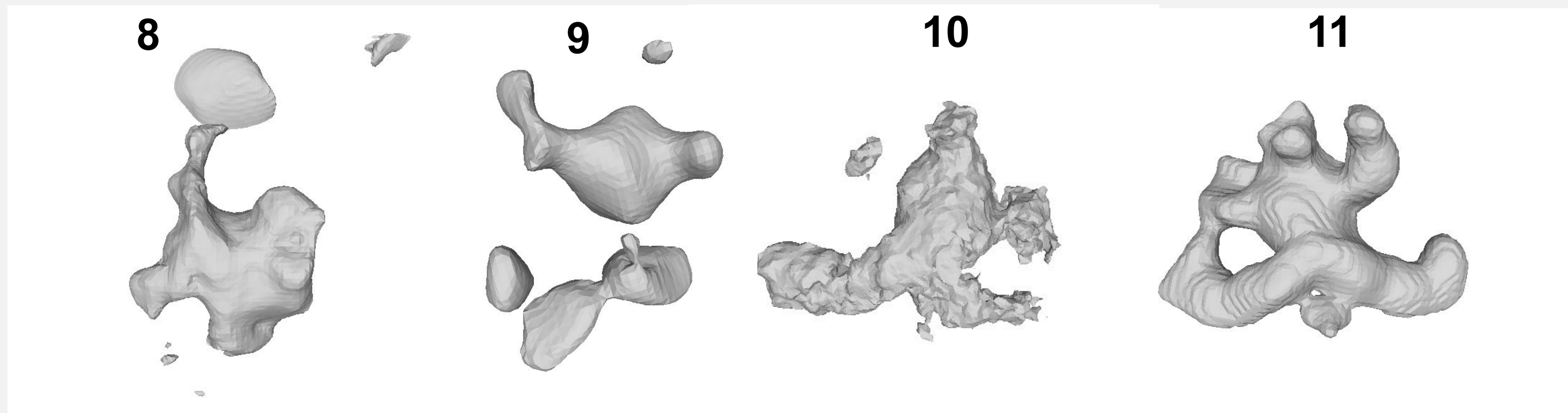
SDF-Overfitting using NN Distance Extractor

We (1) sample random points from the environment of the shape and (2) calculate the distance of that samples to the closest surface of the shape. We use this as input for our (3) SDF overfitting, with the 3d coordinates as input and the distance value as target values. Lastly, we can (4) render the intrinsic 3d shape representation.



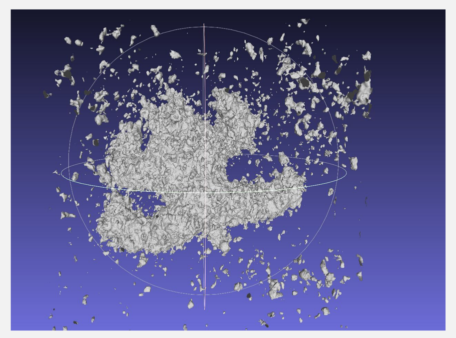
Qualitative Results

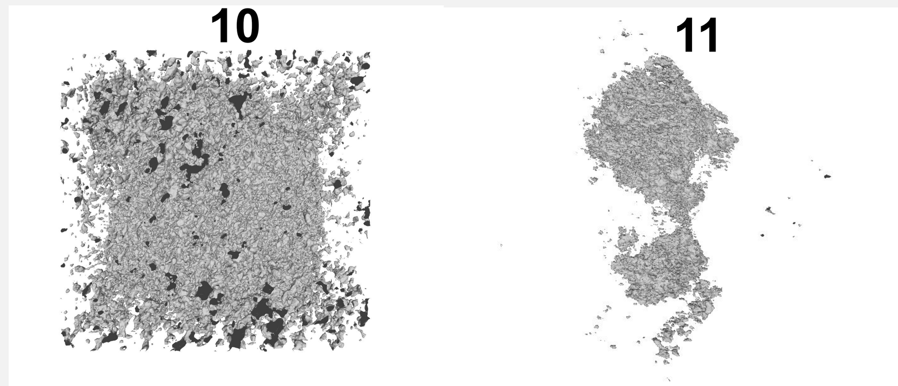
M.-cubes rendering

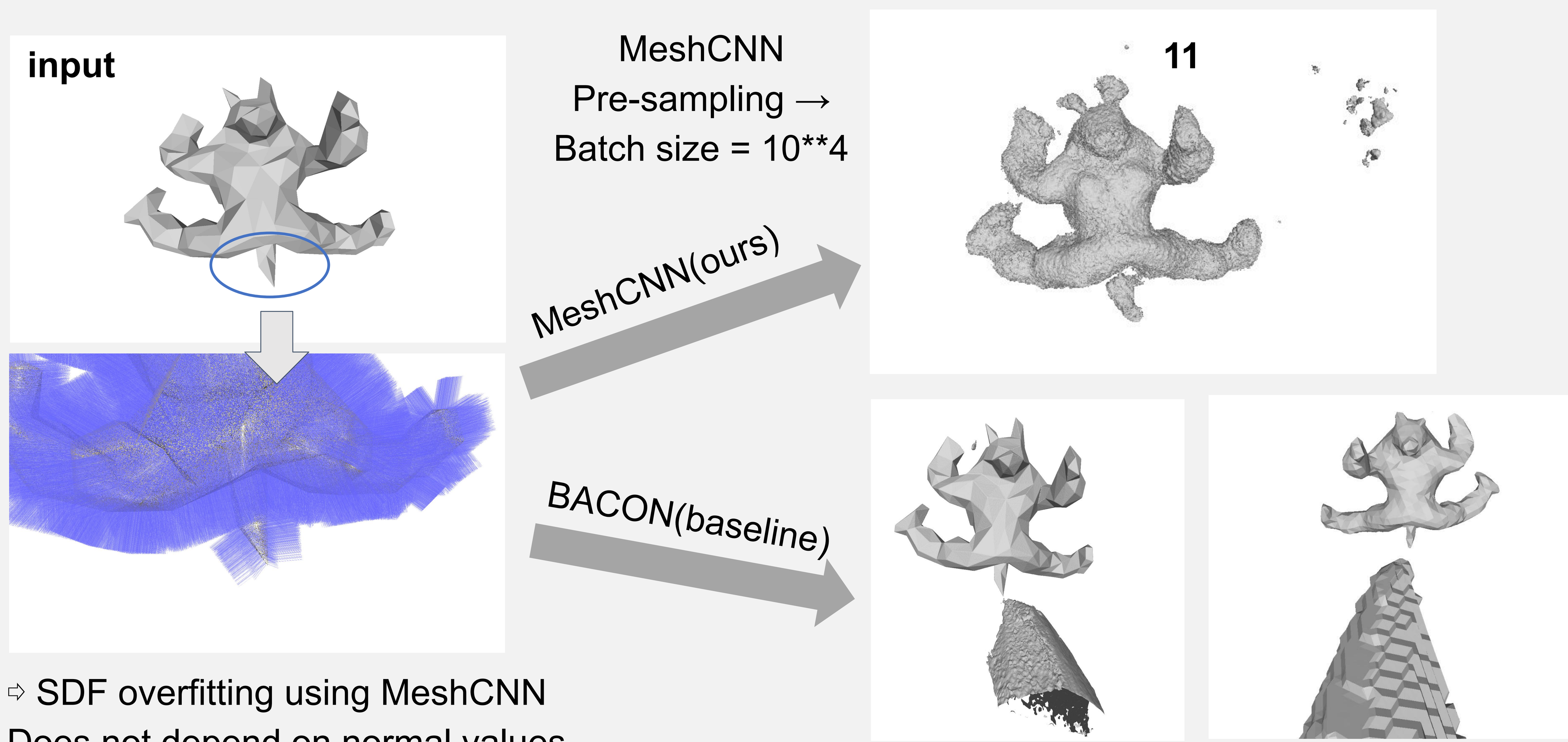


MeshCNN requires a huge amount of resources and overfitting SDF values did not work with a low batch size. This is also the case for the NADE scheme

BACON







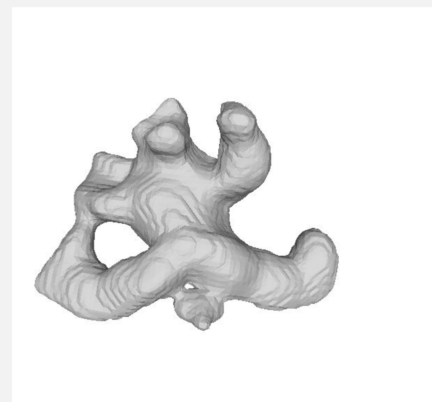
➩ SDF overfitting using MeshCNN
Does not depend on normal values

SDF overfitting M.-cubes rendering

Ablations

Key changes:

1. **MeshCNN Architecture**
2. **Positional Encoding**
3. **No augmentations**
4. **Mesh Normalization**
5. **Deactivated Relu**
6. Loss
7. Freq Tuning Positional Encoding
8. Larger Capacity
9. **Longer Training**
10. **LR adoption**
11. **No Pooling**

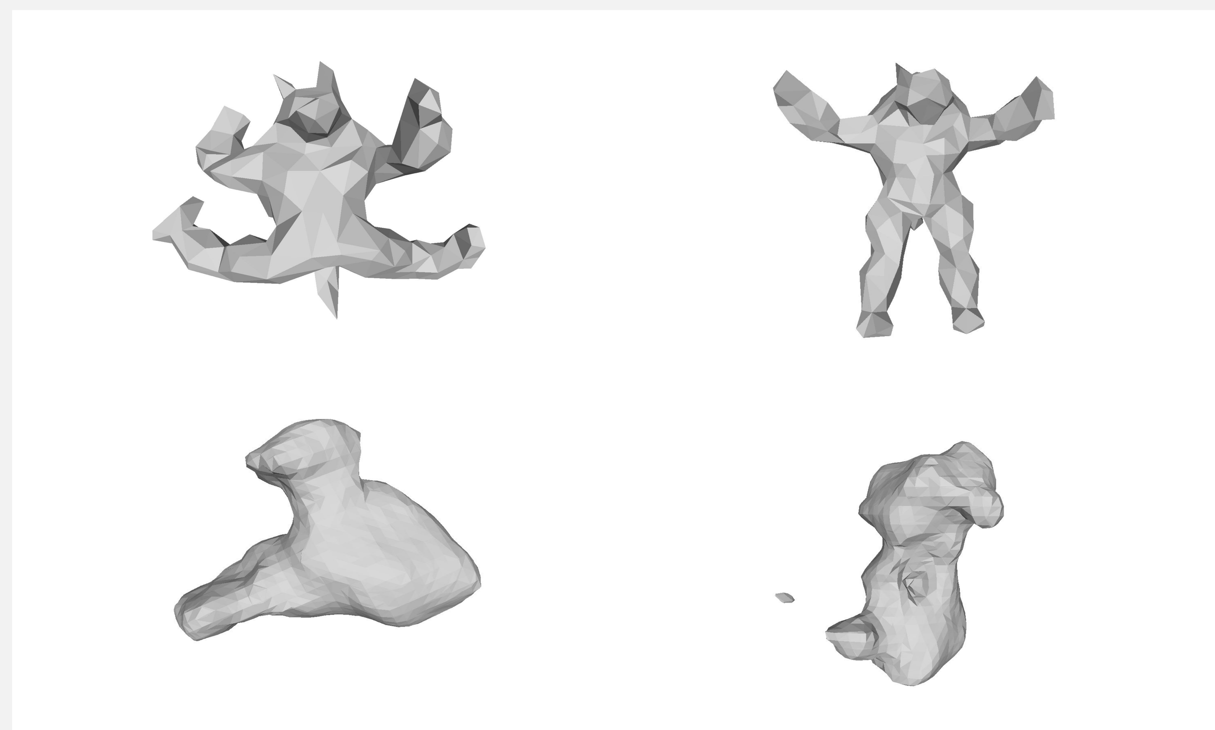


Identifier	MAE e-3⬇	Sign Acc. %⬆	Chamfer ⬇
MLP Baseline	63.00	-	-
MeshCNN - Base	9.67	-	0.0555
Nerf Posi. Encode.	8.48	-	0.0329
No augment.*	6.79	73.80%	0.0296
Mesh Normaliz.	6.27	74.39%	0.0623
Deactivated Relu	6.03	74.81%	0.0523
Combined Loss	7.76	70.36%	0.0594
Sign penalty loss	35.14	71.11%	0.0774
6 frequency encoding	5.86	75.60%	0.0282
Long Train*	5.72	75.53%	0.0324
Larger capacity	5.59	74.47%	0.0309
LR x 16*	4.22	77.81%	0.0328
No pooling	**4.13**	**78.33%**	**0.0276**
Generali.	5.90	75.25%	0.0526

* Training parameter tuning related changes

Generalization

➩ The potential of our MeshCNN based Distance Extractor to process unseen shapes accurately. Trained on 4 similar shapes evaluated in- and out-of-domain.



➔ General Shape and high-level similarities
➔ No details, even in-domain
➔ Extreme long training
➔ Way lower accuracy metrics
➔ Larger training data set needed