DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation

...

Park, J. J., Florence, P., Straub, J., Newcombe, R., & Lovegrove, S. (2019). Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019.

Major Contributions of the Paper

- A novel representation for 3D shapes/objects using the signed distance function (SDF).
- Auto-decoder based generative models for representing 3D shapes.

Representations for 3D Shapes

There are different representation for 3D shapes, each with it's pros and cons:

1. **Point-based**: Primitive representation closely matching raw data.
2. **Mesh-based**: There are many different sub-categories:
    a. Representing shapes with template meshes and deformation factors. [1, 2]
    b. Parameterization of mesh surfaces. [3]
    c. Graph neural network and MeshCNN based representations. [4, 5]
3. **Voxel-based**: Occupancy grids [6], Octree-based [7] and truncated-SDFs [8].

[1] Bagautdinov, Timur, et al. "Modeling facial geometry using compositional vaes." CVPR 2018.
[2] Litany, Or, et al. "Deformable shape completion with graph convolutional autoencoders." CVPR2018.
[3] Groueix, T., et al. "AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation. arXiv 2018." arXiv preprint arXiv:1802.05384 (1802).
[4] Hanocka, Rana, et al. "Meshcnn: a network with an edge." ACM Transactions on Graphics (TOG) 38.4 (2019): 1-12.
[5] Bruna, Joan, et al. "Spectral networks and locally connected networks on graphs." arXiv preprint arXiv:1312.6203 (2013).
[6] Wu, Zhirong, et al. "3d shapenets: A deep representation for volumetric shapes." CVPR 2015.
[7] Tatarchenko, Maxim, et al. "Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs." ICCV 2017.
[8] Curless, Brian, and Marc Levoy. "A volumetric method for building complex models from range images." Proceedings of the 23rd annual conference on Computer graphics and interactive techniques. 1996.

Signed Distance Function (SDFs)

- SDF is a continuous function that, for a given spatial point, returns distance to the closest surface, whose sign encodes whether the point is inside (negative) or outside (positive) of the watertight surface. Mathematically expressed as,

$$SDF(x) = s \text{ where } x \varepsilon R^3 \text{ and } s \varepsilon R,$$

$S = \{x \text{ s.t. } SDF(x) = 0\}$ returns the underlying surface. Using the point set S, we can obtain the underlying mesh using Marching Cubes. [1]

[1] Lorensen, William E., and Harvey E. Cline. "Marching cubes: A high resolution 3D surface construction algorithm." ACM siggraph computer graphics 21.4 (1987): 163-169.

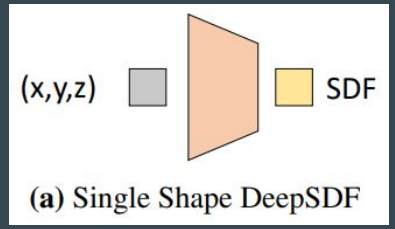
Encoder-Decoder for Learning SDFs

Before looking into auto-decoder SDF, these are some of the common representations that can be used to learn SDF for 3D shapes.

- **Single Shape DeepSDF**: For each shape, a deep network takes as input a point in R^3 and outputs the signed distance. The loss function is defined as follows:

$$L(f_\theta(x),\ s) = | \ clamp(f_\theta(x), \partial) - \ clamp(s, \partial) \ | \ ,$$

where $clamp(s, \partial) = min(\partial , max(\partial, s)$ is introduced to control the distance we maintain from the surface. The major limitation is the need to train one network for one shape.



(a) Single Shape DeepSDF

- **Coded Shape DeepSDF**: For the i^{th} shape, a deep network takes as input a point in R^3 and an additional latent code for that shape (z_i) to output the signed distance. It can be expressed as

$$f_\theta(x, z_i) \approx SDF^i(x),$$

where SDF^i is set of point for which the SDF values are known w.r.t the i^{th} shape. An encoder-decoder network can be used where the encoder takes some input observation to generate z_i and then decoder takes as input the latent code and the query point to output the signed distance. There are two major drawbacks with this approach:

a. It is not robust i.e. this setup does not perform well for new shapes.
b. The design and complexity of the encoder depends on input observation.

Auto-Decoders for representing Shapes

In auto-decoder, there is no encoder to learn latent codes. Thus, the latent codes are dependent on the input set of SDF values for each shape. Let $\mathbf{z}_i$ be the latent code and $X_i = \{(x_j, s_j)\ s.t.\ s_j = SDF^i(x_j)\}$ be the set of SDF observations for the the ith shape then,

$$p(z_i \mid X_i) = p(z_i) \prod_{(x,s)\,\varepsilon\, X_i} p_\theta(s \mid x, z_i), \qquad (1)$$

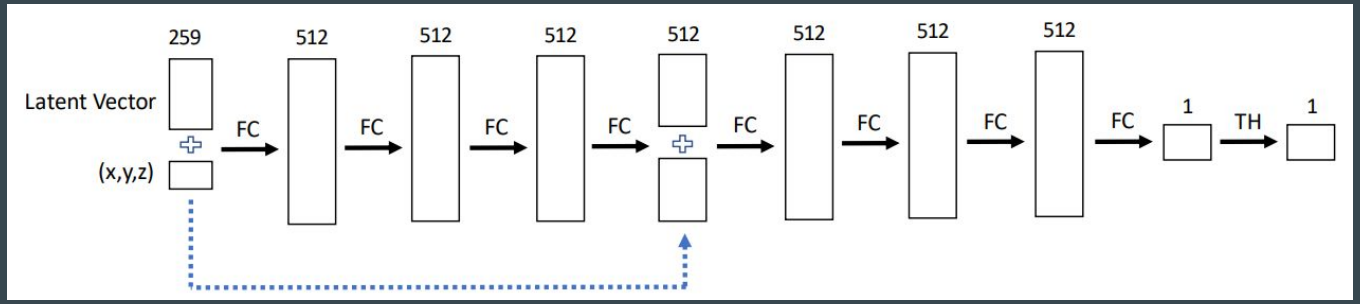
where $p(z_i)$ is initialized as a multivariate gaussian distribution and Θ parameterizes the SDF likelihood which can be expressed as $p_\theta(s \mid x, z_i) = 1 / C\ exp(-L(f_\theta(x), s))$. The goal is to maximize the (1) which can be reformulated by taking the log as,

$$\min_{\theta, z_i, i=\{1..N\}} \sum_{i=1..N} \left(\sum_{j=1..K} L(f_\theta(x_j, z_i), s_j) + 1/\sigma^2 \|z_i\|^2 \right), \qquad (2)$$

At training time, (2) is minimized w.r.t to both z_i and θ simultaneously thereby learning a latent code and training a DeepSDF network to generate SDF. At inference time, the parameters are fixed and the objective function is minimized to learn the optimal $\tilde{z}$

$$\tilde{z} = \operatorname{argmin}_z \left(\sum_{j=1 \ldots K} L\left(f_\theta(x_j, z), s_j \right) + 1/\sigma^2 \|z\|^2 \right),$$

Note the entire formulation is independent of the number of point samples in each X_i.



The above figure gives an architectural overview of the Auto-Decoder DeepSDF Network

Data Preparation

The DeepSDF network is trained using synthetic objects from ShapeNet dataset [1]. To prepare the data, each mesh is normalized to a unit sphere and 500,000 points are sampled with more points sampled near the surface. To obtain SDF values for watertight meshes, for each point the nearest triangle face on the mesh is located.

[1] Chang, Angel X., et al. "Shapenet: An information-rich 3d model repository." arXiv preprint arXiv:1512.03012 (2015).

Experiments And Results

- **Representation of Known Shapes**: The representation power of DeepSDF was tested using shapes present in the training set. The comparison was done based on Chamfer Distance and Earth Mover's Distance between the generated and ground truth shape. The following table summarizes their results:

Method \metric	CD, mean	CD, median	EMD, mean	EMD, median
OGN	0.167	0.127	**0.043**	**0.042**
AtlasNet-Sph.	0.210	0.185	0.046	0.045
AtlasNet-25	0.157	0.140	0.060	0.060
DeepSDF	**0.084**	**0.058**	**0.043**	**0.042**

Table 2: Comparison for representing known shapes (K) for cars trained on ShapeNet. CD = Chamfer Distance (30,000 points) multiplied by 10^3, EMD = Earth Mover's Distance (500 points).

- **Representation of Unknown Shapes**: DeepSDF also boast superior performance in representing shapes that are not present in the training set. The following figure shows some of their results compared against AtlasNet [1]:

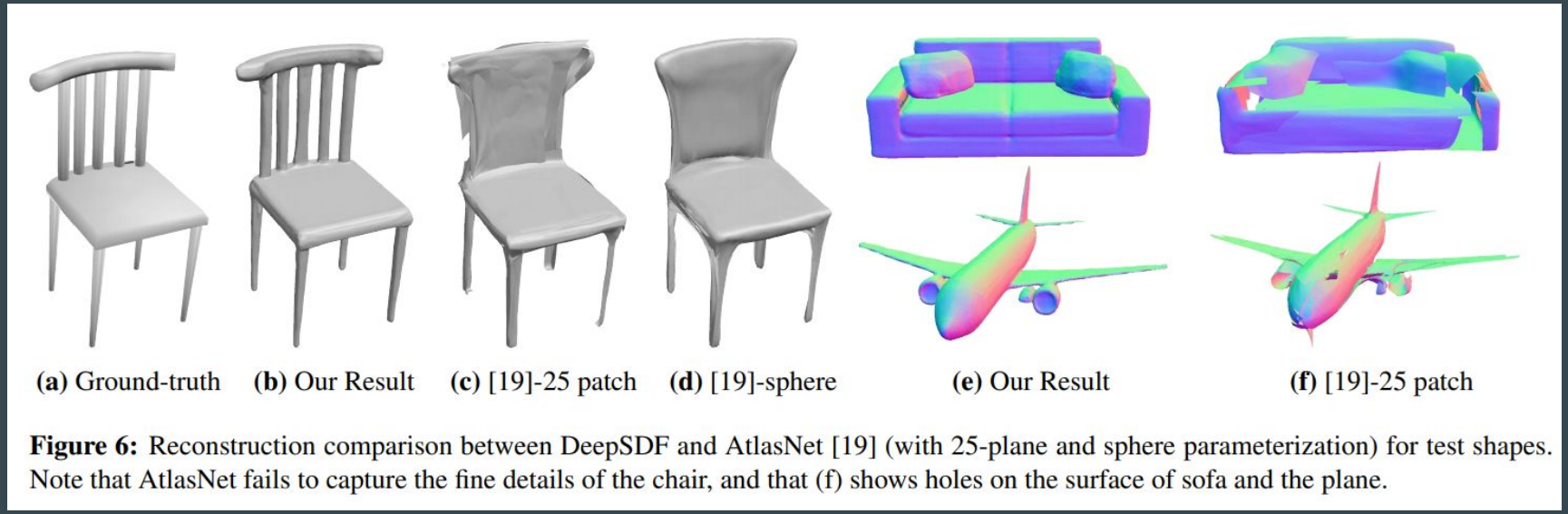
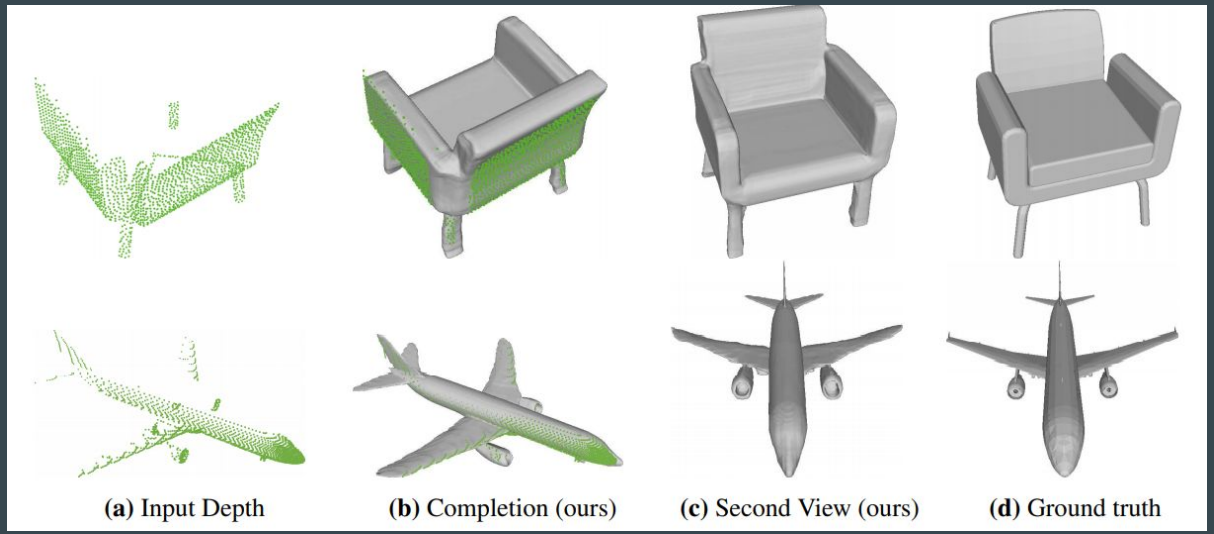


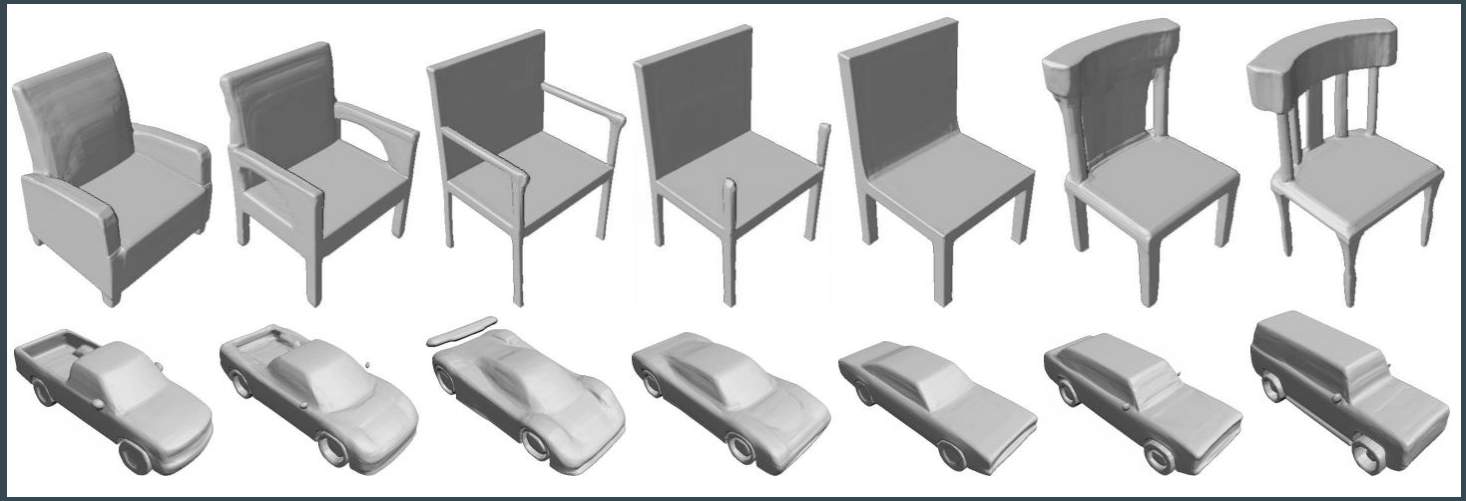
Figure 6: Reconstruction comparison between DeepSDF and AtlasNet [19] (with 25-plane and sphere parameterization) for test shapes. Note that AtlasNet fails to capture the fine details of the chair, and that (f) shows holes on the surface of sofa and the plane.

[1] Groueix, T., et al. "AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation. arXiv 2018." arXiv preprint arXiv:1802.05384 (1802).

- **Shape Completion**: At inference time, learning the optimal latent code and generating surface points is independent of the number of input samples. Thus it can be used for shape completion. In shape completion, the input is normally a depth map from a single view point then the data is prepared by taking two points at distance δ and $-\delta$ from each surface point and then the optimal latent code is determined.



- **Interpolation Across Shapes**: The authors also show that the learnt embeddings are continuous by performing interpolation across two latent codes. The following figure shows the results of interpolation across multiple latent codes.



Conclusion And Future Work

- Unlike Voxel-based and Mesh-based approaches that are compute intensive, DeepSDF are highly efficient as it only uses linear layers and at inference time an infinite set of points can be queried one-by-one to generate a dense surface.
- One major drawback is that the optimal latent code must learnt for each shape at time of inferencing which is time consuming.
- For future work, the inference times can be reduced by using faster optimizers than ADAM.