Neural Scene Representations for 3D Reconstruction

Songyou Peng
ETH Zurich & MPI
10.02.2022
Who Am I?

1992 – 2015: Live and study in China

2015 – 2017: Master in Europe
● Internship at
● Master thesis at

2018 – 2019: Research Engineer in Singapore

2019 – Now: PhD Student at ETH Zurich & MPI
● With Marc Pollefeys and Andreas Geiger
● Internship at

Songyou Peng

https://pengsongyou.github.io/
Agenda

● 3D Scene Representations
● [ECCV’20] Convolutional Occupancy Networks
● [NeurIPS’21] Shape As Points: A Differentiable Poisson Solver
● [arXiv’21]: NICE-SLAM: Neural Implicit Scalable Encoding for SLAM
3D Representations

- Traditional Explicit Representations ⇒ Discrete
Neural Implicit Representations

\[ p \xrightarrow[]{} \text{SDF} \]
Occupancy
Color
Semantics
......
3D Representations

- Traditional Explicit Representations ⇒ Discrete
- Neural Implicit Representations ⇒ Continuous

Mescheder, Oechsle, Niemeyer, Nowozin and Geiger: Occupancy Networks: Learning 3D Reconstruction in Function Space. CVPR, 2019
Limitations

Structure of neural implicit representations:

Mescheder, Oechsle, Niemeyer, Nowozin and Geiger: Occupancy Networks: Learning 3D Reconstruction in Function Space. CVPR, 2019
Limitations

Structure of neural implicit representations:

- Global latent code $\Rightarrow$ overly smooth geometry

Mescheder, Oechsle, Niemeyer, Nowozin and Geiger: Occupancy Networks: Learning 3D Reconstruction in Function Space. CVPR, 2019
Limitations

Structure of neural implicit representations:

- Global latent code $\Rightarrow$ overly smooth geometry
- Fully-connected architecture $\Rightarrow$ no translation equivariance

Mescheder, Oechsle, Niemeyer, Nowozin and Geiger: Occupancy Networks: Learning 3D Reconstruction in Function Space. CVPR, 2019
Limitations

Implicit models work well for **simple objects** but poorly on **complex scenes**:

Mescheder, Oechsle, Niemeyer, Nowozin and Geiger: Occupancy Networks: Learning 3D Reconstruction in Function Space. CVPR, 2019
How to reconstruct large-scale 3D scenes with neural implicit representations?

Convolutional Occupancy Networks
Convolutional Occupancy Networks

Songyou Peng  Michael Niemeyer  Lars Mescheder  Marc Pollefeys  Andreas Geiger
Main Idea
Main Idea

- **3D Volume Encoder**: Use a local PointNet to process input, volumetric feature encoding
Main Idea

- **3D Volume Encoder**: Use a local PointNet to process input, volumetric feature encoding
- **3D Volume Decoder**: Processed by 3D U-Net, query features via trilinear interpolation
- **Occupancy Readout**: Shallow occupancy network $f_\theta(\cdot)$
Main Idea - 2D

- **2D Plane Encoder**: Use a local PointNet to process input, project onto 3-canonical planes
- **2D Plane Decoder**: Processed by U-Net, query features via bilinear interpolation
- **Occupancy Readout**: Shallow occupancy network $f_\theta(\cdot)$
Comparison

Occupancy Networks

- global feature
- heavy FC network
- no translation equivariance

Convolutional Occupancy Networks

- local feature
- shallow FC network
- translation equivariance
Results
Object-Level Reconstruction

Input

ONet

Ours - 2D

Ours - 3D

GT Mesh
Training Speed

![Graph showing the training speed over iterations for different models. The x-axis represents training iterations multiplied by 10K, and the y-axis represents validation IoU. The models compared are PointConv, ONet, Ours-2D (64²), Ours-2D (3 × 64²), and Ours-3D (32³).]
Training Speed

![Graph showing training speed over iterations for different models. The x-axis represents training iterations multiplied by 10K, and the y-axis represents validation IoU. The graph includes lines for PointConv, ONet, Ours-2D (64^2), Ours-2D (3 × 64^2), and Ours-3D (32^3).]
Scene-Level Reconstruction: Synthetic

- Trained and evaluated on synthetic rooms

Input

GT Mesh
Scene-Level Reconstruction: Synthetic

- ONet fails on room-level reconstruction
Scene-Level Reconstruction: Synthetic

- SPSR requires surface normals, output is noisy
Scene-Level Reconstruction: Synthetic

- Our method preserves better details
Large-Scale Reconstruction

Results on Matterport3D

- Fully convolutional model
- Trained on synthetic crops
- Sliding-window evaluation
- Scale to any scene size

Scene size: 15.7m x 12.3m x 4.5m

Our reconstruction output
Large-Scale Reconstruction

Results on Matterport3D

- Fully convolutional model
- Trained on synthetic crops
- Sliding-window evaluation
- Scale to any scene size

Scene size: 15.7m x 12.3m x 4.5m

Our reconstruction output
Conclusions

- ConvONet allows for scaling to large-scale scenes
- ConvONet generalizes well from synthetic to real scenes
- ConvONet trains faster than original ONet

Limitations

- **Slow inference** due to dense grid evaluation
- **Difficult to initialize**
Shape As Points
A Differentiable Poisson Solver

Songyou Peng  Chiyu “Max” Jiang  Yiyi Liao  Michael Niemeyer  Marc Pollefeys  Andreas Geiger
3D Representations

Traditional Explicit Representations

- Discrete
- Fast inference
3D Shape Representations

**Neural Implicit Representations**

- Continuous, watertight
- Slow inference
- Difficult to initialize

Mescheder, Oechsle, Niemeyer, Nowozin and Geiger: Occupancy Networks: Learning 3D Reconstruction in Function Space. CVPR, 2019
3D Shape Representations

Shape As Points (SAP) - Hybrid Representation
+ Discrete (Oriented point clouds) ⇒ Continuous (Implicit indicator grid)
+ Fast inference
+ Easy initialization
Differentiable Poisson Solver

DPSR

in

out
Intuition of Poisson Equation

\[ \nabla^2 \chi := \nabla \cdot \nabla \chi = \nabla \cdot \mathbf{v} \]
Our Poisson Solver

\[ \nabla^2 \chi := \nabla \cdot \nabla \chi = \nabla \cdot \mathbf{v} \]

- **Discretization** allows to invert the divergence operator

\[ \chi = (\nabla^2)^{-1} \nabla \cdot \mathbf{v} \]

- **Spectral methods** to solve the Poisson equation efficiently
  - Derivatives of signals in spectral domain are computed analytically
  - Fast Fourier Transform (FFT) are **highly optimized on GPUs/TPUs**
  - Only **25-line code**

\[
\mathbf{\tilde{v}} = \text{FFT}(\mathbf{v}) \quad \longrightarrow \quad \mathbf{\tilde{\chi}} = \tilde{g}_{\sigma,r}(\mathbf{u}) \odot \frac{\mathbf{i} \mathbf{u} \cdot \mathbf{\tilde{v}}}{-2\pi \|\mathbf{u}\|^2} \quad \longrightarrow \quad \chi' = \text{IFFT}(\mathbf{\tilde{\chi}})
\]
SAP for Optimization-based 3D Reconstruction
Pipeline - Forward Pass

Input an initial oriented point cloud
(noisy / incomplete observations)
Pipeline - Forward Pass
Pipeline - Forward Pass
Pipeline - Forward Pass

\[ p \]

- DPSR
- Marching Cubes
Pipeline - Forward Pass

\[ p \xrightarrow{\text{DPSR}} \text{Marching Cubes} \xrightarrow{} p_{\text{mesh}} \]
Pipeline - Forward Pass

DPSR → Marching Cubes → $p_{\text{mesh}}$ → $\mathcal{L}_{CD}$ → Target
Pipeline - Backward Pass

\[
\frac{\partial \mathcal{L}_{CD}}{\partial \mathbf{p}} = \frac{\partial \mathcal{L}_{CD}}{\partial \mathbf{p}_{\text{mesh}}} \frac{\partial \mathbf{p}_{\text{mesh}}}{\partial \chi} \frac{\partial \chi}{\partial \mathbf{p}}
\]
Pipeline

DPSR → Marching Cubes → Target

Optimize Points and Normals

$\mathcal{L}_{CD}$
Pipeline

Shape-As-Points

DPSR

Marching Cubes

$\mathcal{L}_{CD}$

Target

Optimize Points and Normals
Comparison

Unoriented Point Clouds

GT Mesh
Comparison

Unoriented Point Clouds

Point2Mesh

Runtime: 62 mins
Comparison

Unoriented Point Clouds

IGR

Runtime: 30 mins
Comparison

Unoriented Point Clouds

SAP

Runtime: ~6 mins
Comparison

SPSR

Runtime: ~9 sec

SAP

Runtime: ~6 mins

Kazhdan and Hoppe: Screened Poisson Surface Reconstruction. SIGGRAPH, 2013
Can we further leverage the *differentiability* of the Poisson solver for *deep neural networks*?

SAP for Learning-based 3D Reconstruction
Learning-based Pipeline

Noisy Input

Offsets $f_\theta$

Normals $g_\theta$

Optimize Parameters
Learning-based Pipeline

Noisy Input

Offsets $f_\theta$
Normals $g_\theta$

Optimize Parameters

DPSR
Learning-based Pipeline

Noisy Input

Offsets $f_\theta$
Normals $g_\theta$

Optimize Parameters

DPSR

Indicator Function

PSR

Sample

Ground Truth
Learning-based Pipeline

Noisy Input

Offsets $f_\theta$

Normals $g_\theta$

Optimize Parameters

DPSR

$\mathcal{L}_{\text{DPSR}}$

Indicator Function

PSR

Sample

Ground Truth
Learning-based Pipeline

Noisy Input → DPSR → Marching Cubes → Mesh Output
Inputs

GT Mesh

R2N2
15 ms

AtlasNet
25 ms
Inputs

GT Mesh

ConvONet

327 ms

Ours

64 ms
## Benefit of Geometric Initialization

Chamfer distance over the training process

<table>
<thead>
<tr>
<th>Iterations</th>
<th>10K</th>
<th>50K</th>
<th>100K</th>
<th>200K</th>
<th>Best</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConvONet</td>
<td>0.082</td>
<td>0.058</td>
<td>0.055</td>
<td>0.050</td>
<td>0.044</td>
</tr>
<tr>
<td><strong>Ours</strong></td>
<td><strong>0.041</strong></td>
<td><strong>0.036</strong></td>
<td><strong>0.035</strong></td>
<td><strong>0.034</strong></td>
<td><strong>0.034</strong></td>
</tr>
</tbody>
</table>

SAP converges much faster!
Conclusions

- SAP is **interpretable**, **lightweight** and guarantees **HQ watertight meshes**
- SAP is also **topology agnostic**, enables **fast inference**
- Our Poisson solver is **differentiable** and **GPU-accelerated**

**Limitation**: Cubic memory requirements limits SAP for small scenes
SO WHAT'S NEXT...
Neural Radiance Field (NeRF) **Heat**

**Learning Scene Textures**
Neural Radiance Field (NeRF)

\[
\hat{C}(r) = \sum_{i=1}^{N} T_i (1 - \exp(-\sigma_i \delta_i))c_i , \text{ where } T_i = \exp\left(-\sum_{j=1}^{i-1} \sigma_j \delta_j\right)
\]
KiloNeRF: Speeding up NeRF with Thousands of Tiny MLPs
Christian Reiser, Songyou Peng, Yiyi Liao, Andreas Geiger
ICCV 2021

UNISURF: Unifying Neural Implicit Surfaces and Radiance Fields for Multi-View Reconstruction
Michael Oechsle, Songyou Peng, Andreas Geiger
ICCV 2021 (Oral)
Given an RGB-D sequence, jointly represent geometry & color?

What if the camera poses are also unknown?

Neural Implicit Representations

for

Simultaneous Localization and Mapping (SLAM)
NICE-SLAM
Neural Implicit Scalable Encoding for SLAM

Zihan Zhu\textsuperscript{1,2} * Songyou Peng\textsuperscript{1,3} * Viktor Larsson\textsuperscript{1} Weiwei Xu\textsuperscript{2} Hujun Bao\textsuperscript{2}
Zhaopeng Cui\textsuperscript{2} # Martin R. Oswald\textsuperscript{1,4} Marc Pollefeys\textsuperscript{1,5}

(* equal contribution)

\textsuperscript{1}ETH Zurich  \textsuperscript{2}State Key Lab of CAD&CG, Zhejiang University
\textsuperscript{3}MPI for Intelligent Systems, Tübingen  \textsuperscript{4}University of Amsterdam  \textsuperscript{5}Microsoft

https://pengsongyou.github.io/nice-slam
iMAP

- First dense SLAM system that uses a neural scene representation
- Use a single fully-connected network to represent the entire scene
- Low-quality geometry, fail in larger scenes

Sucar, Liu, Ortiz, Davison: iMAP: Implicit Mapping and Positioning in Real-Time. ICCV 2021
NICE-SLAM

Hierarchical grid-based encoding
High-quality scene geometry & camera tracking on large-scale scene
Local updates -> No forgetting problem for geometry
NICE-SLAM Overview

Input Depth

Input RGB

Hierarchical Feature Grid

Camera Pose
NICE-SLAM Overview

5D Input
Position + Direction

\[(x, y, z, \theta, \phi) \rightarrow \mathcal{F}_\Theta \rightarrow (RGB\sigma) \]

Output
Color + Density
NICE-SLAM Overview
NICE-SLAM Overview
Reformulate the volume rendering equation along each ray as [1]:

\[
\hat{D}^f = \sum_{i=1}^{N} w_i^f d_i, \quad \hat{I} = \sum_{i=1}^{N} w_i^f c_i
\]

\[
w_i^f = o_p \prod_{j=1}^{i-1} (1 - o_p)
\]
NICE-SLAM Overview
NICE-SLAM Overview
NICE-SLAM Overview

Minimize

Input Depth
Reconstruction Loss
Depth Loss
Generated Depth
Photometric Loss
Generated RGB
Input RGB

Mapping

Coarse Occupancy
Tri-linear Interpolation
Fine Level Occupancy

Hierarchical Feature Grid

Tracking

Ray -> Point Sampler
Camera Pose
Results
ScanNet Dataset
<table>
<thead>
<tr>
<th>Method</th>
<th>FLOPs $[\times 10^3] \downarrow$</th>
<th>Tracking [ms] $\downarrow$</th>
<th>Mapping [ms] $\downarrow$</th>
</tr>
</thead>
<tbody>
<tr>
<td>iMAP [42]</td>
<td>443.91</td>
<td>101</td>
<td>448</td>
</tr>
<tr>
<td>NICE-SLAM (Ours)</td>
<td>104.16</td>
<td>47</td>
<td>130</td>
</tr>
</tbody>
</table>

Computations & Runtime
Results

Large Apartment with Multiple Rooms
Conclusion

- Hierarchical Encoding + Implicit decoders -> Scalable scene representations
- Show decent results on large indoor scenes
- Do not have forgetting problems

Limitation

- Geometry & colors are OK, but far from satisfactory
- Camera tracking is worse than traditional methods
- Cannot scale up to outdoor scenes
Thank You!

**ConvONet**
ECCV’20 (Spotlight)

**Shape As Points**
NeurIPS’21 (Oral)

**NICE-SLAM**
arXiv’21

https://pengsongyou.github.io/conv_onet

https://pengsongyou.github.io/sap

https://pengsongyou.github.io/nice-slam