Neural Scene Representations and Differentiable Rendering

Michael Niemeyer

Autonomous Vision Group
MPI for Intelligent Systems and University of Tübingen

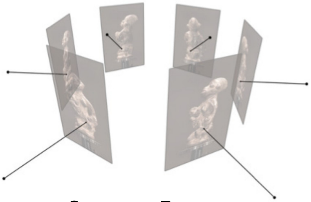
Neural Scene Representations for 3D Reconstruction

Traditional 3D Reconstruction Pipeline

Input Images

Traditional 3D Reconstruction Pipeline

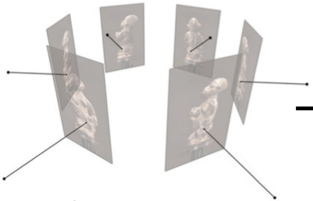
Input Images



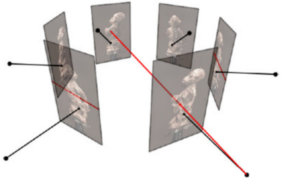
Camera Poses

Traditional 3D Reconstruction Pipeline

Input Images



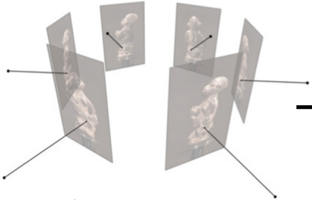
Camera Poses



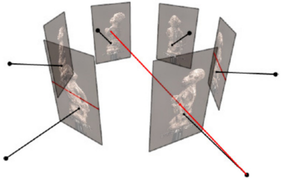
Dense Correspondences

Traditional 3D Reconstruction Pipeline

Input Images



Camera Poses

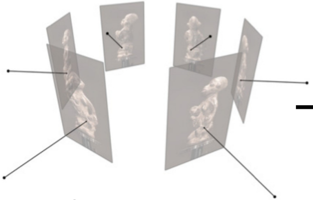


Dense Correspondences

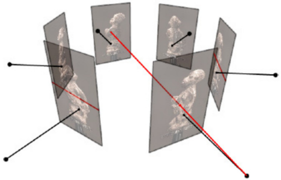
Depth Maps

Traditional 3D Reconstruction Pipeline

Input Images

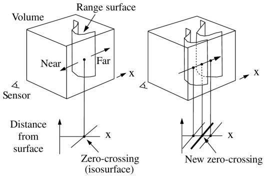


Camera Poses



Dense Correspondences

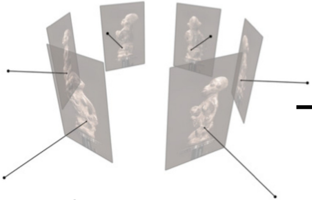
Depth Maps

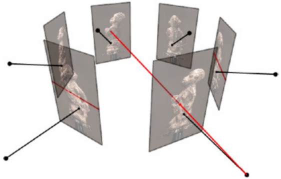


Depth Map Fusion

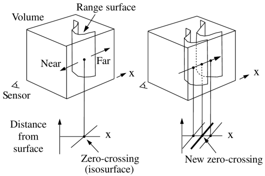
Traditional 3D Reconstruction Pipeline


Input Images

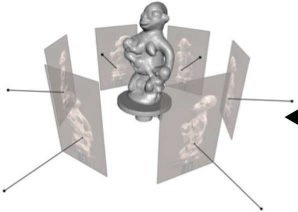

Camera Poses


Dense Correspondences


Depth Maps

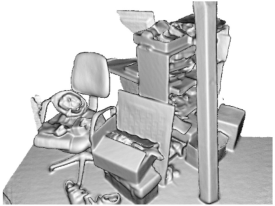


Depth Map Fusion

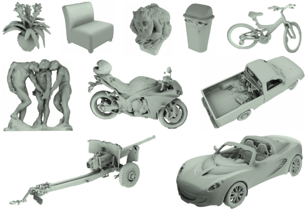

3D Reconstruction

Can we **learn** 3D reconstruction **from data**?

3D Datasets and Repositories



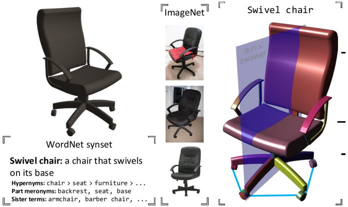
[Newcombe et al., 2011]



[Choi et al., 2011]

[Dai et al., 2017]

[Wu et al., 2015]

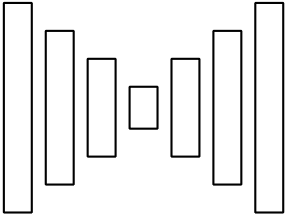


[Chang et al., 2015]

[Chang et al., 2017]

3D Reconstruction from a 2D Image

Input Images



Neural Network

3D Reconstruction

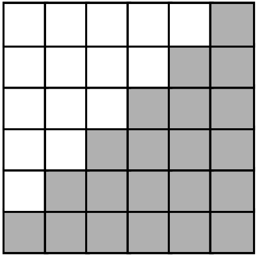
What is a good output representation?

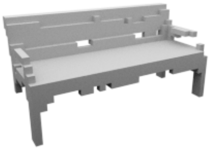
Learning-based 3D Reconstruction

Voxels:

- **Discretization** of 3D space into grid
- Easy to process with neural networks
- Cubic memory $O(n^3) \Rightarrow$ limited resolution
- Manhattan world bias

[Maturana et al., IROS 2015]



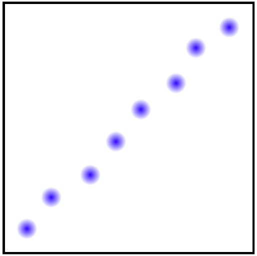


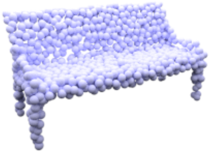
Learning-based 3D Reconstruction

Points:

- **Discretization** of surface into 3D points
- Does not model connectivity / topology
- Limited number of points
- Global shape description

[Fan et al., CVPR 2017]

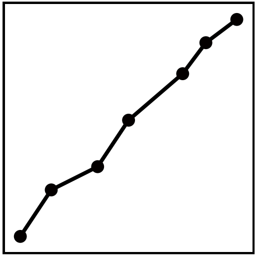




Learning-based 3D Reconstruction

Meshes:

- **Discretization** into vertices and faces
- Limited number of vertices / granularity
- Requires class-specific template – or –
- Leads to self-intersections

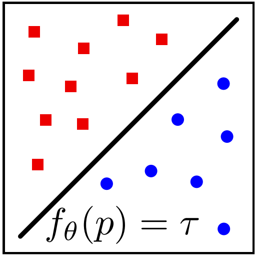


[Groueix et al., CVPR 2018]

Learning-based 3D Reconstruction

This work:

- Implicit representation $\Rightarrow$ **No discretization**
- Arbitrary topology & resolution
- Low memory footprint
- Not restricted to specific class



Occupancy Networks

Key Idea:

- Do not represent 3D shape explicitly

Occupancy Networks

Key Idea:

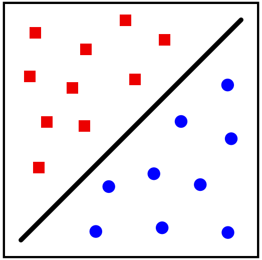
- Do not represent 3D shape explicitly
- Instead, consider surface **implicitly** as **decision boundary** of a non-linear classifier:

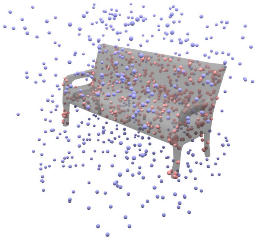
$$f_\theta : \mathbb{R}^3 \times \mathcal{X} \rightarrow [0, 1]$$

3D Location

Condition (eg, Image)

Occupancy Probability





Occupancy Networks

Key Idea:

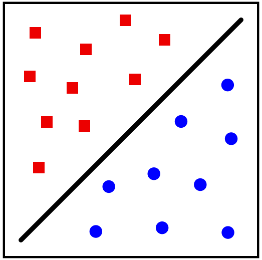
- Do not represent 3D shape explicitly
- Instead, consider surface **implicitly** as **decision boundary** of a non-linear classifier:

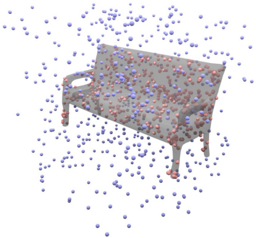
$$f_\theta : \mathbb{R}^3 \times \mathcal{X} \to [0, 1]$$

3D Location — Condition (eg, Image) — Occupancy Probability

Remarks:

- The function f_θ models an **occupancy field**





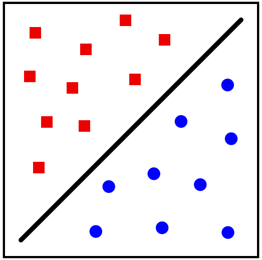
Mescheder, Oechsle, Niemeyer, Nowozin and Geiger: Occupancy Networks: Learning 3D Reconstruction in Function Space. CVPR, 2019.

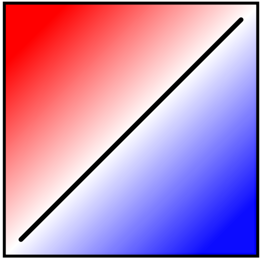
Occupancy Networks

Key Idea:

- Do not represent 3D shape explicitly
- Instead, consider surface **implicitly** as **decision boundary** of a non-linear classifier:

$$f_\theta : \underbrace{\mathbb{R}^3}_{\text{3D Location}} \times \underbrace{\mathcal{X}}_{\text{Condition (eg, Image)}} \to \underbrace{[0,1]}_{\text{Occupancy Probability}}$$



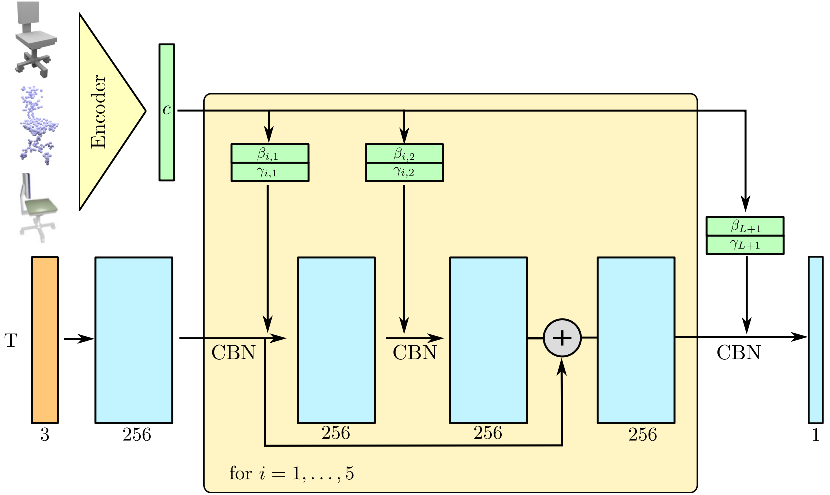


Remarks:

- The function f_θ models an **occupancy field**
- Also possible: **signed distance field** [Park et al., 2019]

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

Network Architecture



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

Training Objective

Occupancy Network:

$$\mathcal{L}(\theta, \psi) = \sum_{j=1}^{K} \text{BCE}(f_\theta(p_{ij}, z_i), o_{ij})$$

- K: Randomly sampled 3D points ($K = 2048$)
- BCE: Cross-entropy loss

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

Training Objective

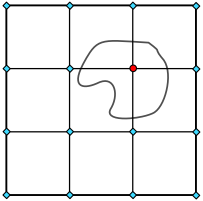
Variational Occupancy Encoder:

$$\mathcal{L}(\theta, \psi) = \sum_{j=1}^{K} \mathrm{BCE}(f_\theta(p_{ij}, z_i), o_{ij}) + KL\left[q_\psi(z|(p_{ij}, o_{ij})_{j=1:K}) \,\|\, p_0(z)\right]$$

- K: Randomly sampled 3D points ($K = 2048$)
- BCE: Cross-entropy loss
- q_ψ: Encoder

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

Occupancy Networks

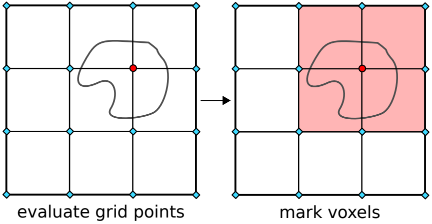


Multiresolution IsoSurface Extraction (MISE):

- Build octree by incrementally querying the occupancy network

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

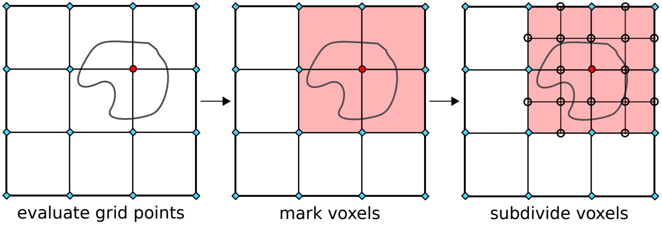
Occupancy Networks



Multiresolution IsoSurface Extraction (MISE):

- Build octree by incrementally querying the occupancy network

Occupancy Networks

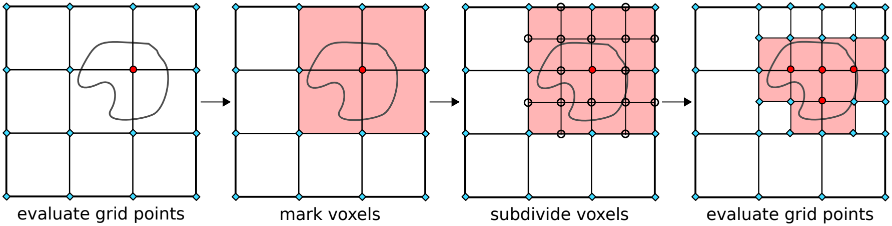


Multiresolution IsoSurface Extraction (MISE):

- Build octree by incrementally querying the occupancy network

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

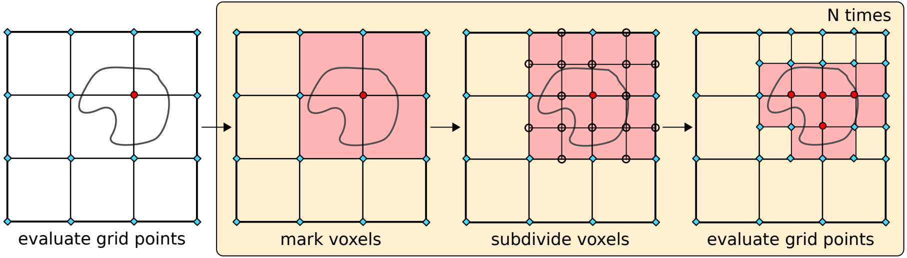
Occupancy Networks



Multiresolution IsoSurface Extraction (MISE):

- Build octree by incrementally querying the occupancy network

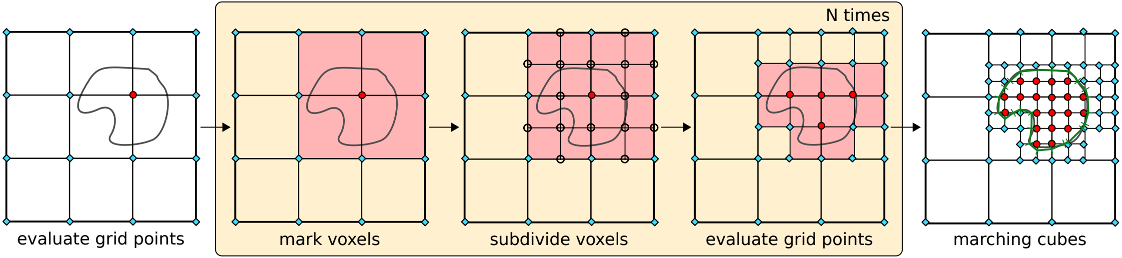
Occupancy Networks



Multiresolution IsoSurface Extraction (MISE):

- Build octree by incrementally querying the occupancy network

Occupancy Networks

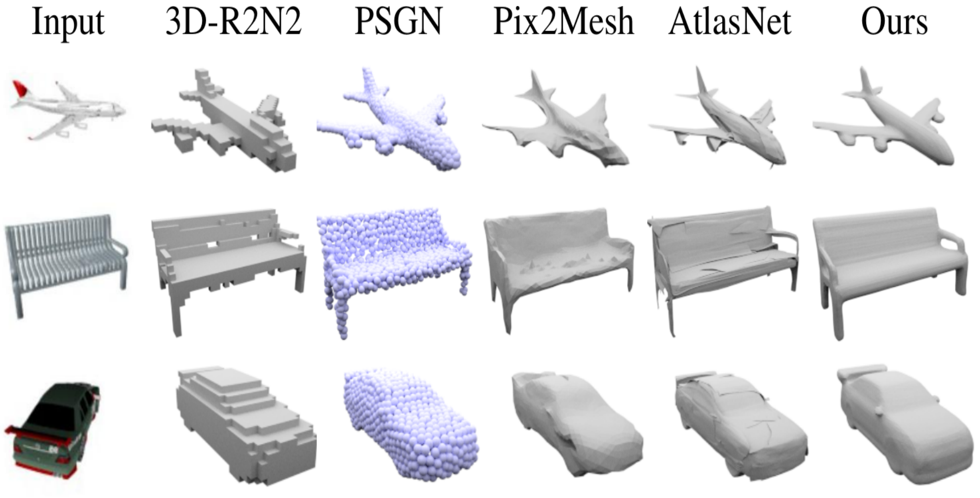


Multiresolution IsoSurface Extraction (MISE):

- Build octree by incrementally querying the occupancy network
- Extract triangular mesh using marching cubes algorithm (1-3 seconds in total)

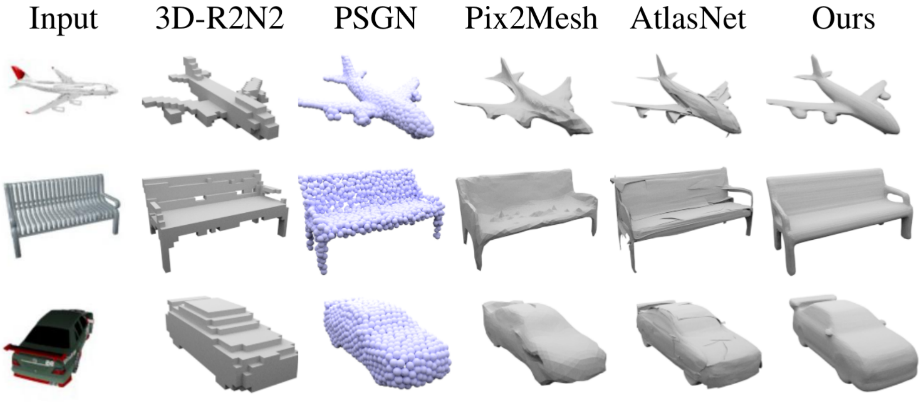
Mescheder, Oechsle, Niemeyer, Nowozin and Geiger: Occupancy Networks: Learning 3D Reconstruction in Function Space. CVPR, 2019.

Results



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

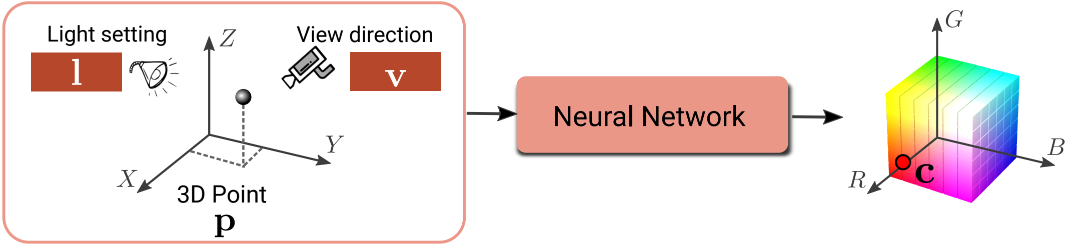
Results



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

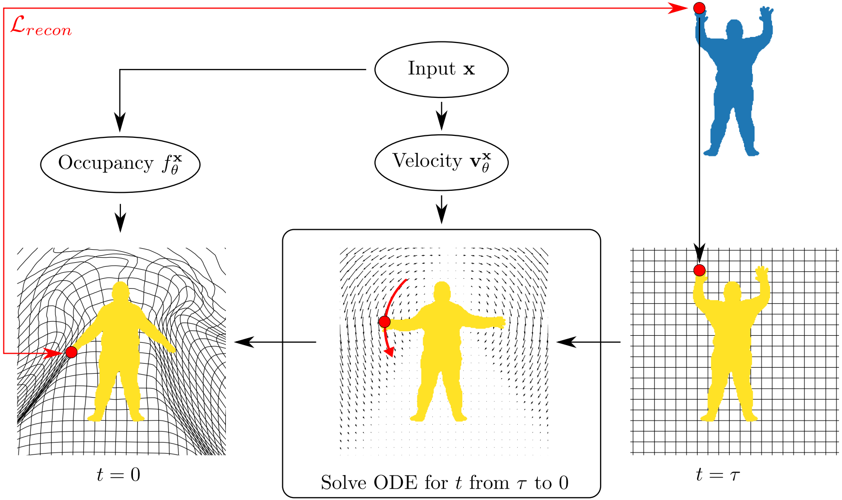
Applications

Appearance



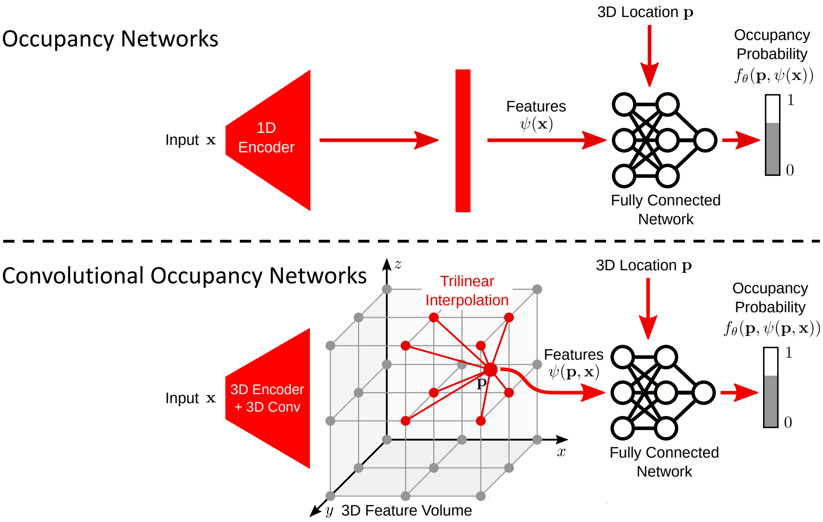
Oechsle, Niemeyer, Mescheder, Strauss and Geiger: Learning Implicit Surface Light Fields. 3DV, 2020.

Motion



$\mathcal{L}_{recon}$
Input $\mathbf{x}$
Occupancy $f_\theta^{\mathbf{x}}$
Velocity $\mathbf{v}_\theta^{\mathbf{x}}$
$t = 0$
Solve ODE for t from τ to 0
$t = \tau$

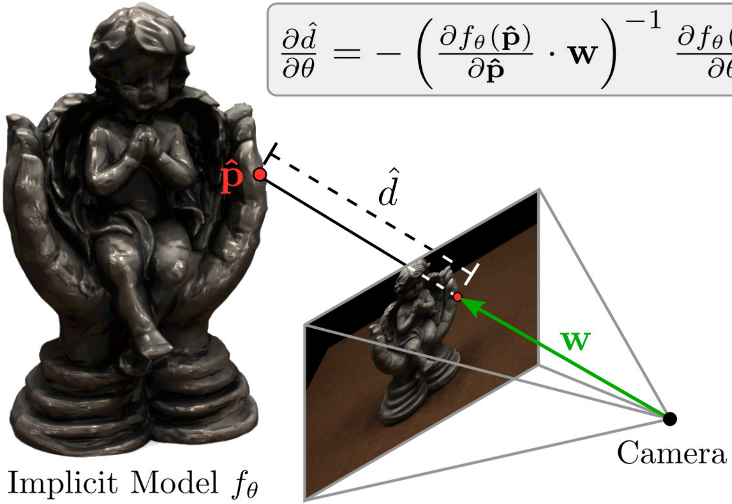
3D Scenes



Peng, Niemeyer, Mescheder, Pollefeys and Geiger: Convolutional Occupancy Networks. ECCV, 2020.

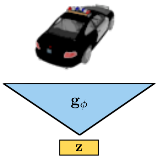
Differentiable Rendering

$$\frac{\partial \hat{d}}{\partial \theta} = -\left(\frac{\partial f_\theta(\hat{\mathbf{p}})}{\partial \hat{\mathbf{p}}} \cdot \mathbf{w}\right)^{-1} \frac{\partial f_\theta(\hat{\mathbf{p}})}{\partial \theta}$$

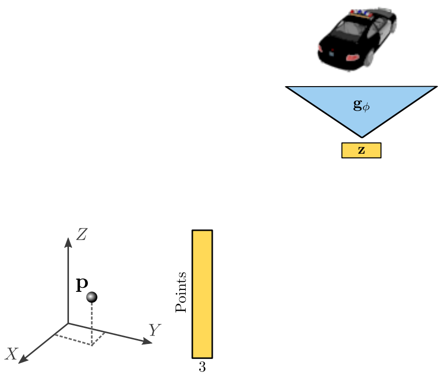


Differentiable Surface Rendering

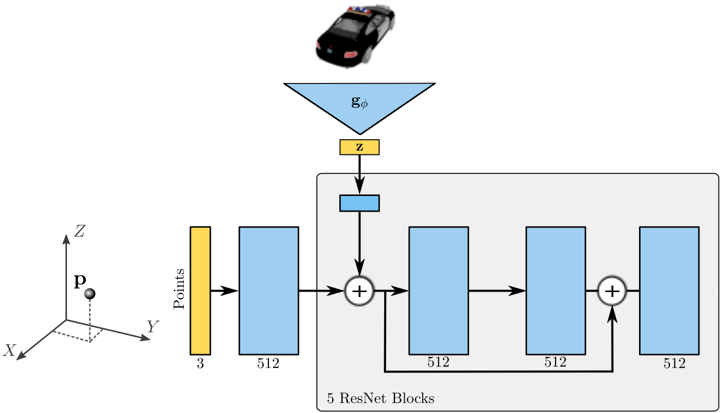
DVR: Differentiable Volumetric Rendering



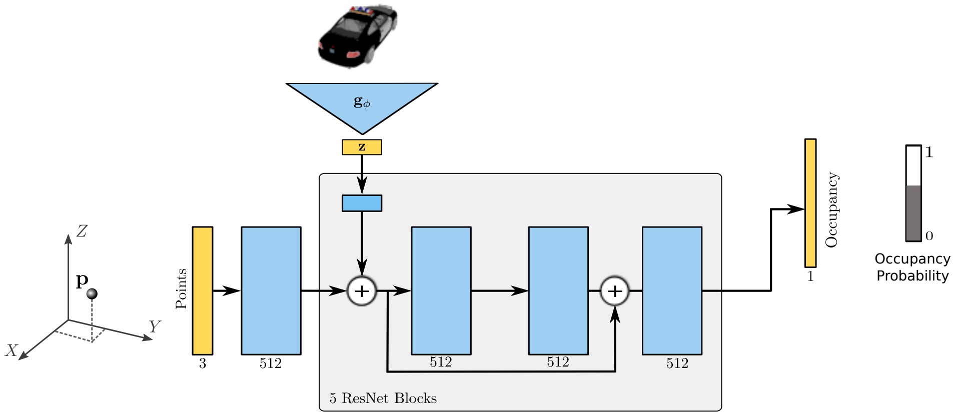
DVR: Differentiable Volumetric Rendering



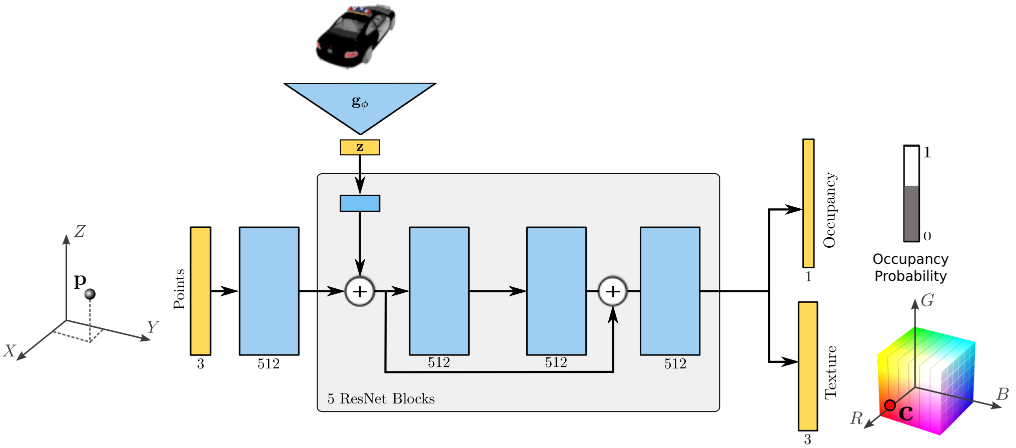
DVR: Differentiable Volumetric Rendering



DVR: Differentiable Volumetric Rendering



DVR: Differentiable Volumetric Rendering

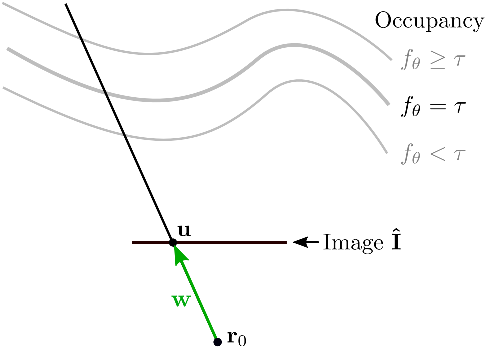


Forward Pass
(Rendering)

DVR: Differentiable Volumetric Rendering

Forward Pass:

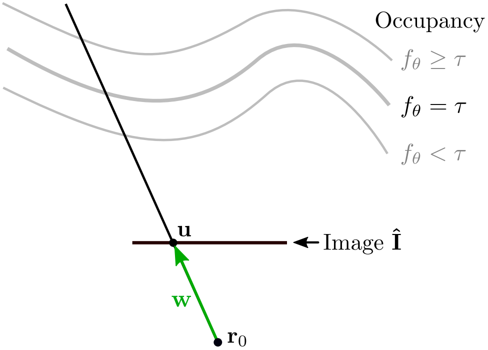
- For all pixels $\mathbf{u}$



DVR: Differentiable Volumetric Rendering

Forward Pass:

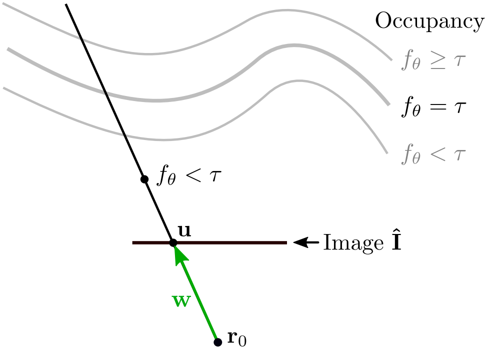
- For all pixels $\mathbf{u}$
- Find surface point $\hat{\mathbf{p}}$ along ray $\mathbf{w}$ via ray marching and root finding (secant method)



DVR: Differentiable Volumetric Rendering

Forward Pass:

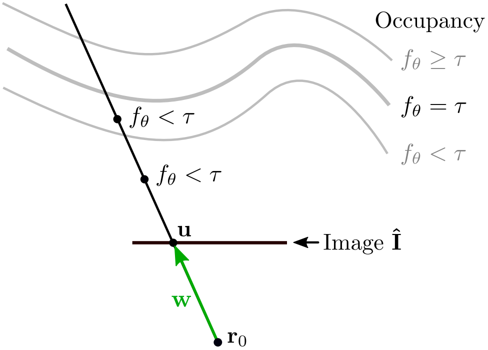
- For all pixels $\mathbf{u}$
- Find surface point $\hat{\mathbf{p}}$ along ray $\mathbf{w}$ via ray marching and root finding (secant method)



DVR: Differentiable Volumetric Rendering

Forward Pass:

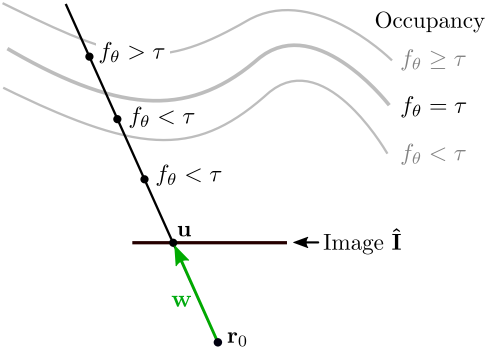
- For all pixels $\mathbf{u}$
- Find surface point $\hat{\mathbf{p}}$ along ray $\mathbf{w}$ via ray marching and root finding (secant method)



DVR: Differentiable Volumetric Rendering

Forward Pass:

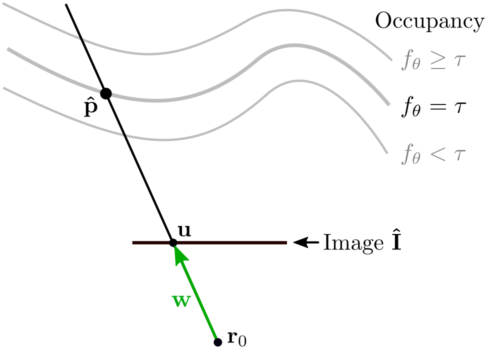
- For all pixels $\mathbf{u}$
- Find surface point $\hat{\mathbf{p}}$ along ray $\mathbf{w}$ via ray marching and root finding (secant method)



DVR: Differentiable Volumetric Rendering

Forward Pass:

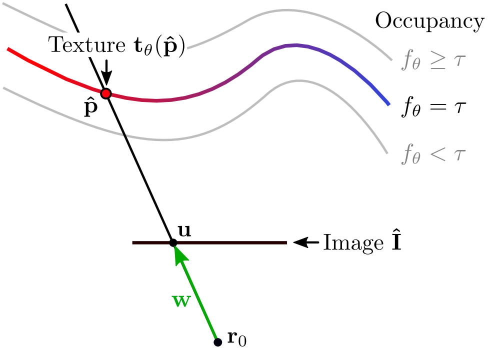
- For all pixels $\mathbf{u}$
- Find surface point $\hat{\mathbf{p}}$ along ray $\mathbf{w}$ via ray marching and root finding (secant method)



DVR: Differentiable Volumetric Rendering

Forward Pass:

- For all pixels $\mathbf{u}$
- Find surface point $\hat{\mathbf{p}}$ along ray $\mathbf{w}$ via ray marching and root finding (secant method)
- Evaluate texture field $\mathbf{t}_\theta(\hat{\mathbf{p}})$ at $\hat{\mathbf{p}}$

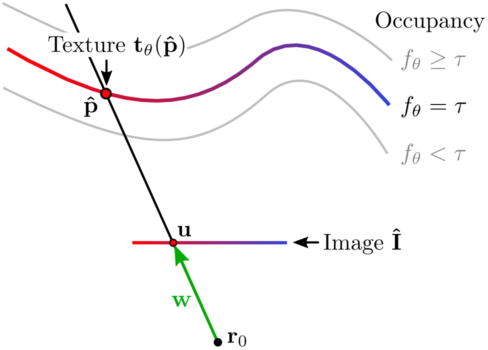


Niemeyer, Mescheder, Oechsle and Geiger: Differentiable Volumetric Rendering: Learning Implicit 3D Representations without 3D Supervision. CVPR, 2020.

DVR: Differentiable Volumetric Rendering

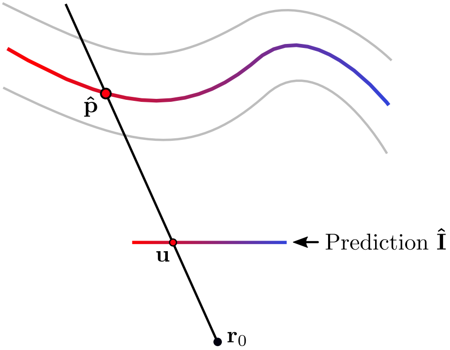
Forward Pass:

- For all pixels $\mathbf{u}$
- Find surface point $\hat{\mathbf{p}}$ along ray $\mathbf{w}$ via ray marching and root finding (secant method)
- Evaluate texture field $\mathbf{t}_\theta(\hat{\mathbf{p}})$ at $\hat{\mathbf{p}}$
- Insert color $\mathbf{t}_\theta(\hat{\mathbf{p}})$ at pixel $\mathbf{u}$



DVR: Differentiable Volumetric Rendering

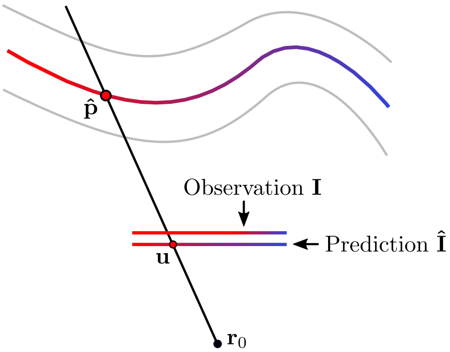
Backward Pass:



DVR: Differentiable Volumetric Rendering

Backward Pass:

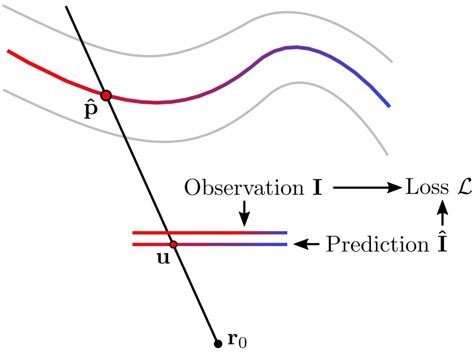
- Image Observation $\mathbf{I}$



DVR: Differentiable Volumetric Rendering

Backward Pass:

- Image Observation $\mathbf{I}$
- Loss $\mathcal{L}(\hat{\mathbf{I}}, \mathbf{I}) = \sum_{\mathbf{u}} \|\hat{\mathbf{I}}_{\mathbf{u}} - \mathbf{I}_{\mathbf{u}}\|$



Niemeyer, Mescheder, Oechsle and Geiger: Differentiable Volumetric Rendering: Learning Implicit 3D Representations without 3D Supervision. CVPR, 2020.

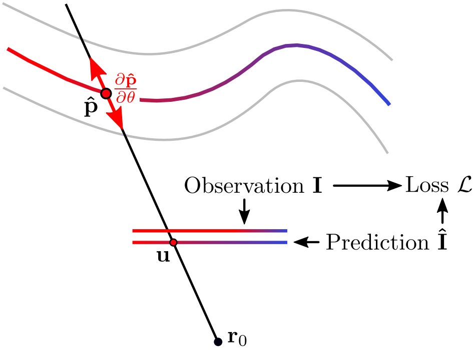
DVR: Differentiable Volumetric Rendering

Backward Pass:

- Image Observation $\mathbf{I}$
- Loss $\mathcal{L}(\hat{\mathbf{I}}, \mathbf{I}) = \sum_{\mathbf{u}} \|\hat{\mathbf{I}}_{\mathbf{u}} - \mathbf{I}_{\mathbf{u}}\|$
- Gradient of loss function:

$$\frac{\partial \mathcal{L}}{\partial \theta} = \sum_{\mathbf{u}} \frac{\partial \mathcal{L}}{\partial \hat{\mathbf{I}}_{\mathbf{u}}} \cdot \frac{\partial \hat{\mathbf{I}}_{\mathbf{u}}}{\partial \theta}$$

$$\frac{\partial \hat{\mathbf{I}}_{\mathbf{u}}}{\partial \theta} = \frac{\partial \mathbf{t}_\theta(\hat{\mathbf{p}})}{\partial \theta} + \frac{\partial \mathbf{t}_\theta(\hat{\mathbf{p}})}{\partial \hat{\mathbf{p}}} \cdot {\color{red}\frac{\partial \hat{\mathbf{p}}}{\partial \theta}}$$



DVR: Differentiable Volumetric Rendering

Backward Pass:

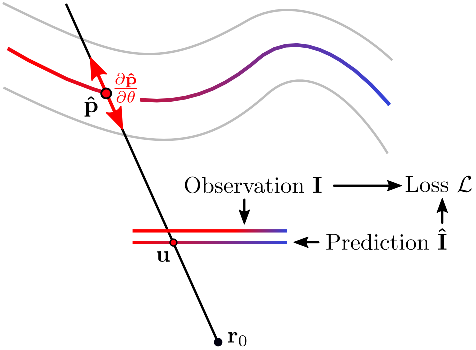
- Image Observation $\mathbf{I}$
- Loss $\mathcal{L}(\hat{\mathbf{I}}, \mathbf{I}) = \sum_{\mathbf{u}} \|\hat{\mathbf{I}}_{\mathbf{u}} - \mathbf{I}_{\mathbf{u}}\|$
- Gradient of loss function:

$$\frac{\partial \mathcal{L}}{\partial \theta} = \sum_{\mathbf{u}} \frac{\partial \mathcal{L}}{\partial \hat{\mathbf{I}}_{\mathbf{u}}} \cdot \frac{\partial \hat{\mathbf{I}}_{\mathbf{u}}}{\partial \theta}$$

$$\frac{\partial \hat{\mathbf{I}}_{\mathbf{u}}}{\partial \theta} = \frac{\partial \mathbf{t}_\theta(\hat{\mathbf{p}})}{\partial \theta} + \frac{\partial \mathbf{t}_\theta(\hat{\mathbf{p}})}{\partial \hat{\mathbf{p}}} \cdot \frac{\partial \hat{\mathbf{p}}}{\partial \theta}$$

- Differentiation of $f_\theta(\hat{\mathbf{p}}) = \tau$ yields:

$$\frac{\partial \hat{\mathbf{p}}}{\partial \theta} = -\mathbf{w} \left(\frac{\partial f_\theta(\hat{\mathbf{p}})}{\partial \hat{\mathbf{p}}} \cdot \mathbf{w} \right)^{-1} \frac{\partial f_\theta(\hat{\mathbf{p}})}{\partial \theta}$$



DVR: Differentiable Volumetric Rendering

Backward Pass:

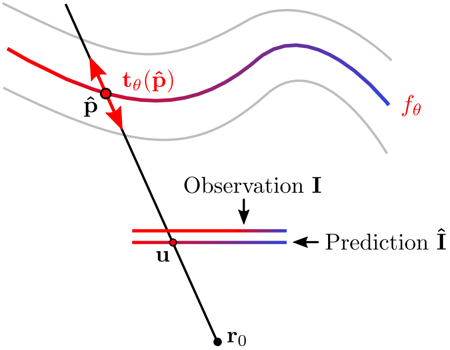
- Image Observation $\mathbf{I}$
- Loss $\mathcal{L}(\hat{\mathbf{I}}, \mathbf{I}) = \sum_{\mathbf{u}} \|\hat{\mathbf{I}}_{\mathbf{u}} - \mathbf{I}_{\mathbf{u}}\|$
- Gradient of loss function:

$$\frac{\partial \mathcal{L}}{\partial \theta} = \sum_{\mathbf{u}} \frac{\partial \mathcal{L}}{\partial \hat{\mathbf{I}}_{\mathbf{u}}} \cdot \frac{\partial \hat{\mathbf{I}}_{\mathbf{u}}}{\partial \theta}$$

$$\frac{\partial \hat{\mathbf{I}}_{\mathbf{u}}}{\partial \theta} = \frac{\partial \mathbf{t}_\theta(\hat{\mathbf{p}})}{\partial \theta} + \frac{\partial \mathbf{t}_\theta(\hat{\mathbf{p}})}{\partial \hat{\mathbf{p}}} \cdot \frac{\partial \hat{\mathbf{p}}}{\partial \theta}$$

- Differentiation of $f_\theta(\hat{\mathbf{p}}) = \tau$ yields:

$$\frac{\partial \hat{\mathbf{p}}}{\partial \theta} = -\mathbf{w} \left(\frac{\partial f_\theta(\hat{\mathbf{p}})}{\partial \hat{\mathbf{p}}} \cdot \mathbf{w} \right)^{-1} \frac{\partial f_\theta(\hat{\mathbf{p}})}{\partial \theta}$$



DVR: Differentiable Volumetric Rendering

Backward Pass:

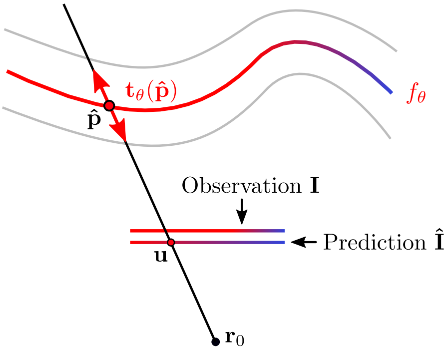
- Image Observation $\mathbf{I}$
- Loss $\mathcal{L}(\hat{\mathbf{I}}, \mathbf{I}) = \sum_{\mathbf{u}} \|\hat{\mathbf{I}}_{\mathbf{u}} - \mathbf{I}_{\mathbf{u}}\|$
- Gradient of loss function:

$$\frac{\partial \mathcal{L}}{\partial \theta} = \sum_{\mathbf{u}} \frac{\partial \mathcal{L}}{\partial \hat{\mathbf{I}}_{\mathbf{u}}} \cdot \frac{\partial \hat{\mathbf{I}}_{\mathbf{u}}}{\partial \theta}$$

$$\frac{\partial \hat{\mathbf{I}}_{\mathbf{u}}}{\partial \theta} = \frac{\partial \mathbf{t}_\theta(\hat{\mathbf{p}})}{\partial \theta} + \frac{\partial \mathbf{t}_\theta(\hat{\mathbf{p}})}{\partial \hat{\mathbf{p}}} \cdot \frac{\partial \hat{\mathbf{p}}}{\partial \theta}$$

- Differentiation of $f_\theta(\hat{\mathbf{p}}) = \tau$ yields:

$$\frac{\partial \hat{\mathbf{p}}}{\partial \theta} = -\mathbf{w} \left(\frac{\partial f_\theta(\hat{\mathbf{p}})}{\partial \hat{\mathbf{p}}} \cdot \mathbf{w} \right)^{-1} \frac{\partial f_\theta(\hat{\mathbf{p}})}{\partial \theta}$$



DVR: Differentiable Volumetric Rendering

Backward Pass:

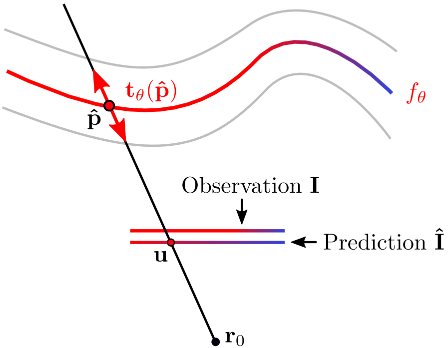
- Image Observation $\mathbf{I}$
- Loss $\mathcal{L}(\hat{\mathbf{I}}, \mathbf{I}) = \sum_{\mathbf{u}} \|\hat{\mathbf{I}}_{\mathbf{u}} - \mathbf{I}_{\mathbf{u}}\|$
- Gradient of loss function:

$$\frac{\partial \mathcal{L}}{\partial \theta} = \sum_{\mathbf{u}} \frac{\partial \mathcal{L}}{\partial \hat{\mathbf{I}}_{\mathbf{u}}} \cdot \frac{\partial \hat{\mathbf{I}}_{\mathbf{u}}}{\partial \theta}$$

$$\frac{\partial \hat{\mathbf{I}}_{\mathbf{u}}}{\partial \theta} = \frac{\partial \mathbf{t}_\theta(\hat{\mathbf{p}})}{\partial \theta} + \frac{\partial \mathbf{t}_\theta(\hat{\mathbf{p}})}{\partial \hat{\mathbf{p}}} \cdot \frac{\partial \hat{\mathbf{p}}}{\partial \theta}$$

- Differentiation of $f_\theta(\hat{\mathbf{p}}) = \tau$ yields:

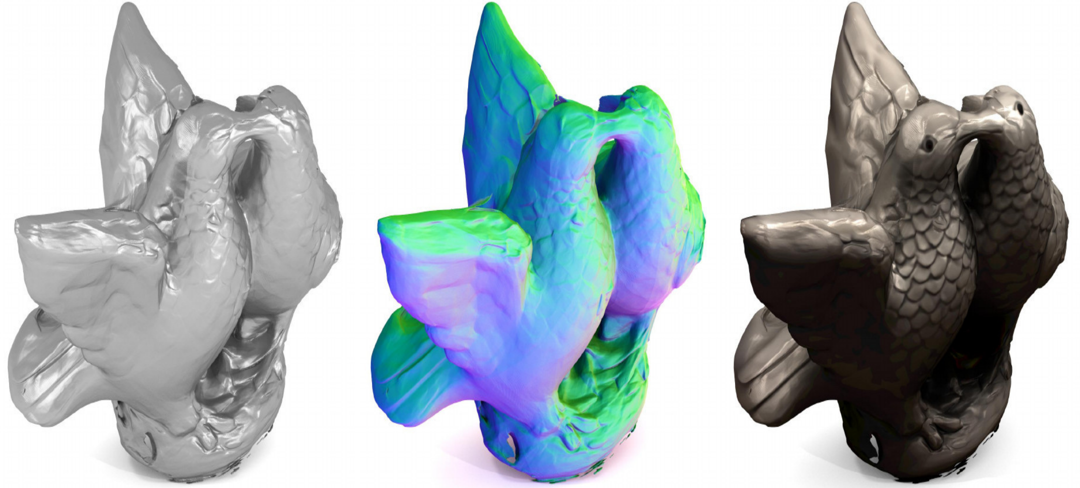
$$\frac{\partial \hat{\mathbf{p}}}{\partial \theta} = -\mathbf{w} \left(\frac{\partial f_\theta(\hat{\mathbf{p}})}{\partial \hat{\mathbf{p}}} \cdot \mathbf{w} \right)^{-1} \frac{\partial f_\theta(\hat{\mathbf{p}})}{\partial \theta}$$



$\Rightarrow$ **Analytic solution** and **no need** for storing **intermediate results**

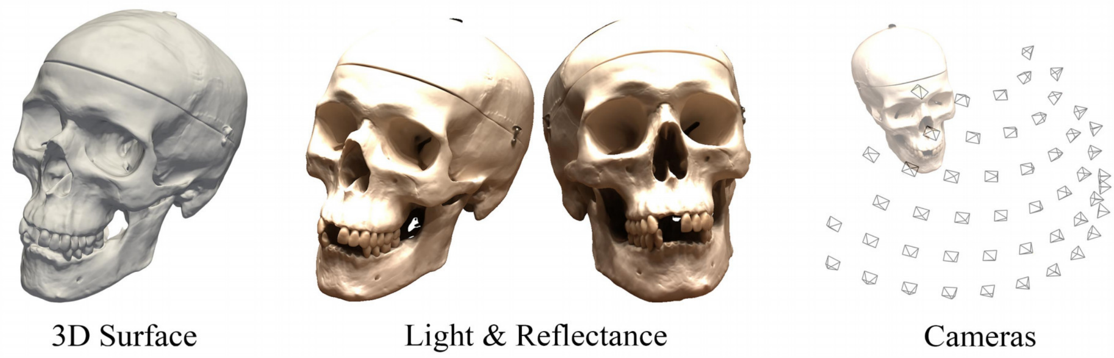
Results

DVR allows for 3D reconstruction from multi-view images of real scenes



Niemeyer, Mescheder, Oechsle and Geiger: Differentiable Volumetric Rendering: Learning Implicit 3D Representations without 3D Supervision. CVPR, 2020.

Implicit Differentiable Renderer



Related work by Lipman et al.:

- Condition on surface normal and view direction for **view-dependent appearance**
- Optimize geometry, appearance and **camera poses**

Yariv, Atzmon and Lipman: Multiview Neural Surface Reconstruction by Disentangling Geometry and Appearance. NeurIPS, 2020.

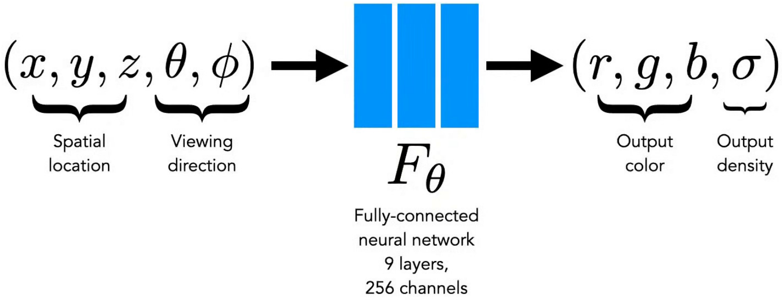
Differentiable Volume Rendering

Novel View Synthesis

- **Task:** Given a set of images of a scene (left), render novel viewpoints (right)

Mildenhall, Srinivasan, Tancik, Barron, Ramamoorthi and Ng: NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. ECCV, 2020.

NeRF: Representing Scenes as Neural Radiance Fields



- Vanilla **ReLU MLP** that maps from **location/view direction to color/density**
- **Density** σ describes how solid/transparent a 3D point is (can model, e.g., fog)
- Conditioning on view direction allows for modeling **view-dependent effects**

Mildenhall, Srinivasan, Tancik, Barron, Ramamoorthi and Ng: NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. ECCV, 2020.

Volume Rendering

Rendering model for ray r(t) = o + td:

$$C \approx \sum_{i=1}^{N} T_i \alpha_i c_i$$

weights

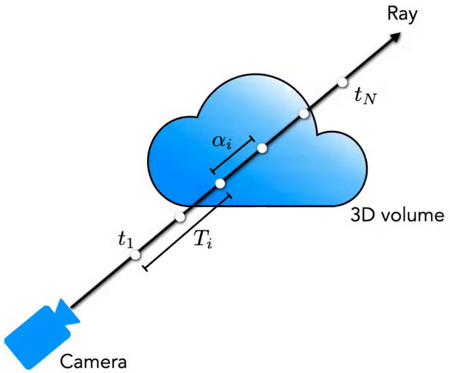
colors

How much light is blocked earlier along ray:

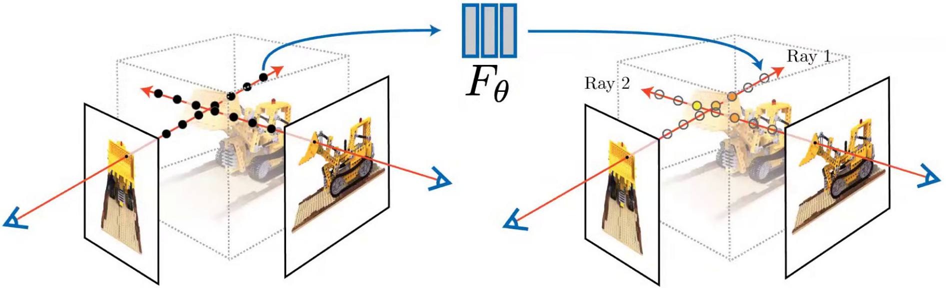
$$T_i = \prod_{j=1}^{i-1} (1 - \alpha_j)$$

How much light is contributed by ray segment *i*:

$$\alpha_i = 1 - e^{-\sigma_i \delta t_i}$$



NeRF Training



$$\min_{\theta} \sum_i ||\text{render}_i(F_\theta) - I_i||^2$$

- Shoot ray, render ray to pixel, minimize **reconstruction error** via backpropagation

Mildenhall, Srinivasan, Tancik, Barron, Ramamoorthi and Ng: NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. ECCV, 2020.

Fourier Features

NeRF (Naive)

NeRF (with positional encoding)

- Essential trick: Compute **positional encoding** for input point $\mathbf{x}$ and direction $\mathbf{d}$

Tancik et al.: Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. NeurIPS, 2020.

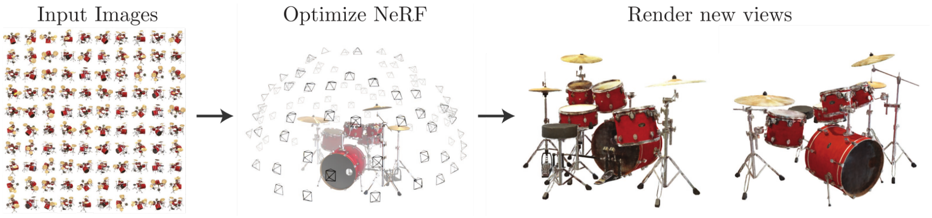
Results

NeRF achieves impressive view synthesis:

Tancik et al.: Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. NeurIPS, 2020.

Scaling to Real-World Scenarios

Scaling to Real-World Scenarios



- In NeRF, **many input images** are assumed to be given

Niemeyer, Barron, Mildenhall, Sajjadi, Geiger, Radwan: RegNeRF: Regularizing Neural Radiance Fields for View Synthesis from Sparse Inputs. CVPR, 2022.

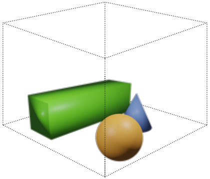
Scaling to Real-World Scenarios

- In the real world, we often have only **sparse inputs**

Niemeyer, Barron, Mildenhall, Sajjadi, Geiger, Radwan: RegNeRF: Regularizing Neural Radiance Fields for View Synthesis from Sparse Inputs. CVPR, 2022.

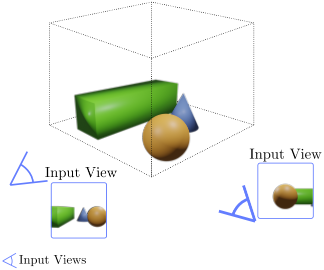
How can we make NeRF work
for sparse input scenarios?

RegNeRF

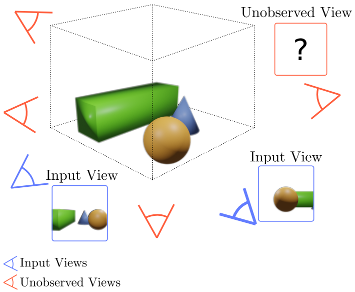


Niemeyer, Barron, Mildenhall, Sajjadi, Geiger, Radwan: RegNeRF: Regularizing Neural Radiance Fields for View Synthesis from Sparse Inputs. CVPR, 2022.

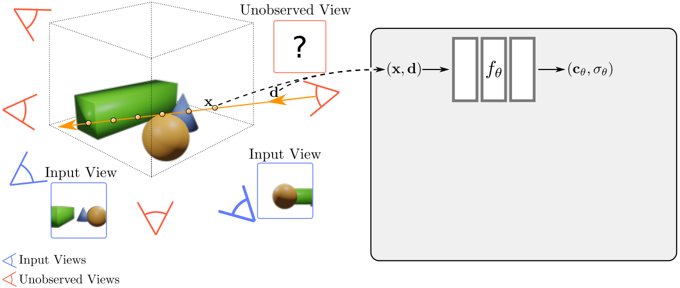
RegNeRF



RegNeRF

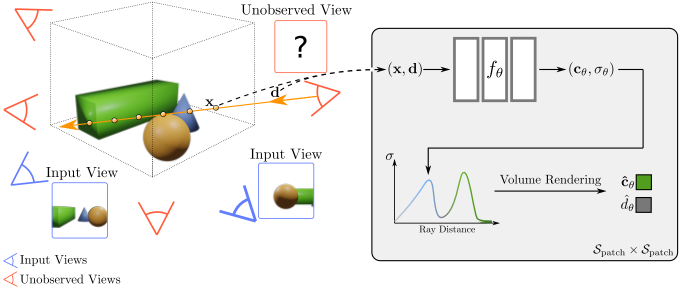

Unobserved View
?
Input View
Input View
Input Views
Unobserved Views

RegNeRF



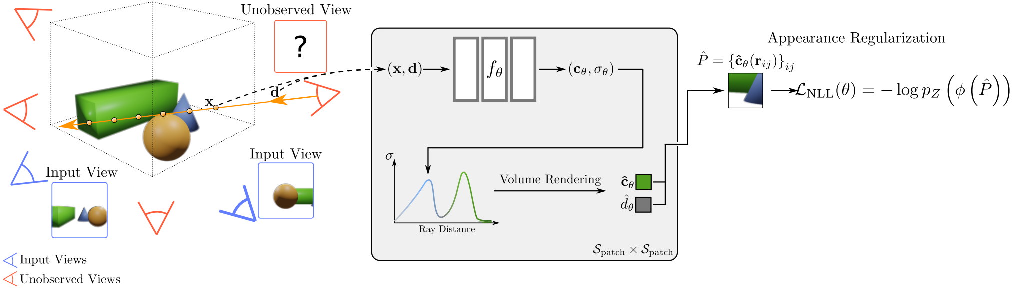
Niemeyer, Barron, Mildenhall, Sajjadi, Geiger, Radwan: RegNeRF: Regularizing Neural Radiance Fields for View Synthesis from Sparse Inputs. CVPR, 2022.

RegNeRF



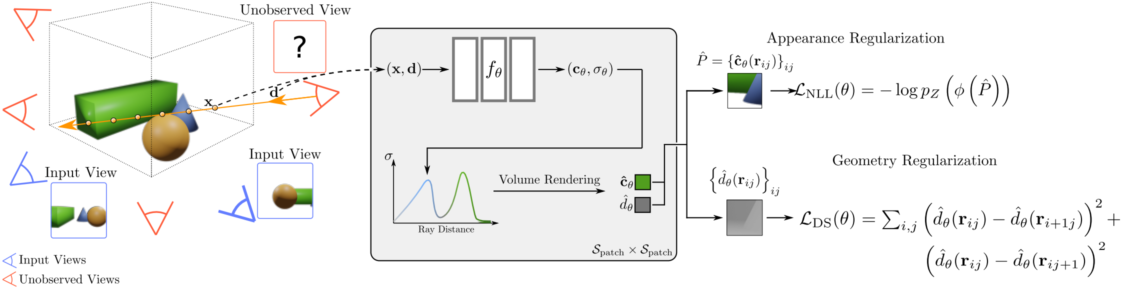
Niemeyer, Barron, Mildenhall, Sajjadi, Geiger, Radwan: RegNeRF: Regularizing Neural Radiance Fields for View Synthesis from Sparse Inputs. CVPR, 2022.

RegNeRF



Niemeyer, Barron, Mildenhall, Sajjadi, Geiger, Radwan: RegNeRF: Regularizing Neural Radiance Fields for View Synthesis from Sparse Inputs. CVPR, 2022.

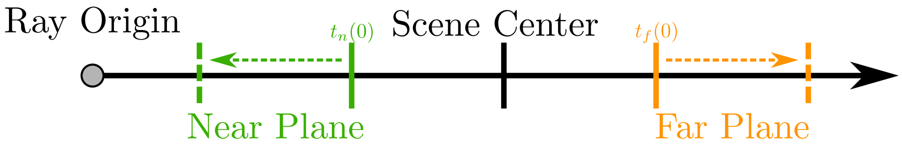
RegNeRF



Niemeyer, Barron, Mildenhall, Sajjadi, Geiger, Radwan: RegNeRF: Regularizing Neural Radiance Fields for View Synthesis from Sparse Inputs. CVPR, 2022.

RegNeRF

We perform **scene space annealing** over first iterations to avoid degenerate solutions:



$$t_n(i) = t_m + (t_n - t_m)\eta(i)$$

$$t_f(i) = t_m + (t_f - t_m)\eta(i)$$

$$\eta(i) = \min\left(\max\left(\frac{i}{N_t}, p_s\right), 1\right)$$

Niemeyer, Barron, Mildenhall, Sajjadi, Geiger, Radwan: RegNeRF: Regularizing Neural Radiance Fields for View Synthesis from Sparse Inputs. CVPR, 2022.

RegNeRF

In summary, the **key ideas** of RegNeRF are

1. Regularizing the geometry prediction of unseen viewpoints
2. Regularizing the appearance prediction of unseen viewpoints
3. Performing scene space annealing over the first iterations

Niemeyer, Barron, Mildenhall, Sajjadi, Geiger, Radwan: RegNeRF: Regularizing Neural Radiance Fields for View Synthesis from Sparse Inputs. CVPR, 2022.

RegNeRF

Comparison mipNeRF and RegNeRF for **three input images**:

Niemeyer, Barron, Mildenhall, Sajjadi, Geiger, Radwan: RegNeRF: Regularizing Neural Radiance Fields for View Synthesis from Sparse Inputs. CVPR, 2022.

RegNeRF

Comparison mipNeRF and RegNeRF for **three input images**:

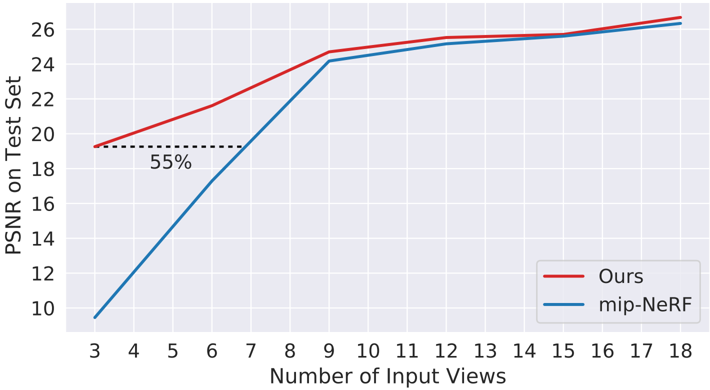
Niemeyer, Barron, Mildenhall, Sajjadi, Geiger, Radwan: RegNeRF: Regularizing Neural Radiance Fields for View Synthesis from Sparse Inputs. CVPR, 2022.

RegNeRF

Comparison mipNeRF and RegNeRF for **three input images**:

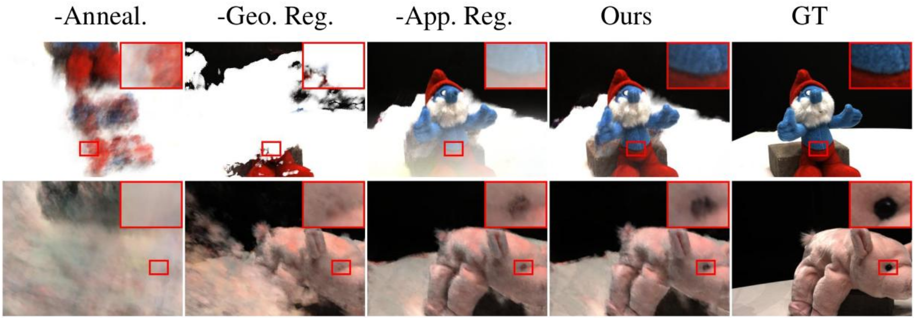
RegNeRF

Performance wrt. the number of input views:



Niemeyer, Barron, Mildenhall, Sajjadi, Geiger, Radwan: RegNeRF: Regularizing Neural Radiance Fields for View Synthesis from Sparse Inputs. CVPR, 2022.

Ablation Study



Niemeyer, Barron, Mildenhall, Sajjadi, Geiger, Radwan: RegNeRF: Regularizing Neural Radiance Fields for View Synthesis from Sparse Inputs. CVPR, 2022.

Summary

Summary

- Occupancy Networks: neural fields are a powerful 3D representation
- DVR: neural fields can be inferred from 2D supervision via differentiable rendering
- RegNeRF: Regularization allows to scale to real-world scenarios

Limitations

- Incorporating compositional scene understanding
- Scaling to more cluttered real-world scenes
- Faster training and inference for sparse input scenarios