CVPR SEATTLE WASHINGTON JUNE 16-18 2020

DIST: Rendering Deep Implicit Signed Distance Function with Differentiable Sphere Tracing

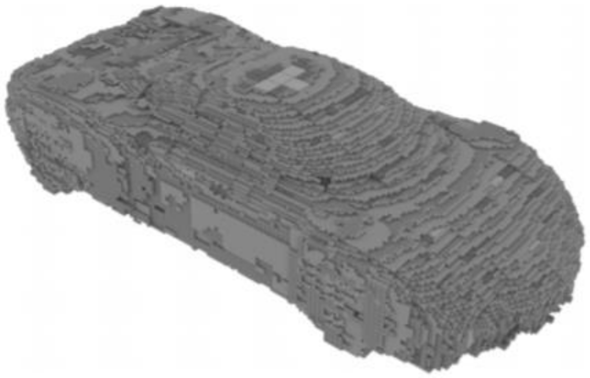
Shaohui Liu, Yinda Zhang, Songyou Peng, Boxin Shi, Marc Pollefeys, Zhaopeng Cui

ETH zürich Google Tsinghua University Peking University Microsoft

Deep Implicit Signed Distance Functions (DeepSDF)

- Infinite-Resolution
- Lightweight

Effective compared to
voxels, point clouds and triangular meshes



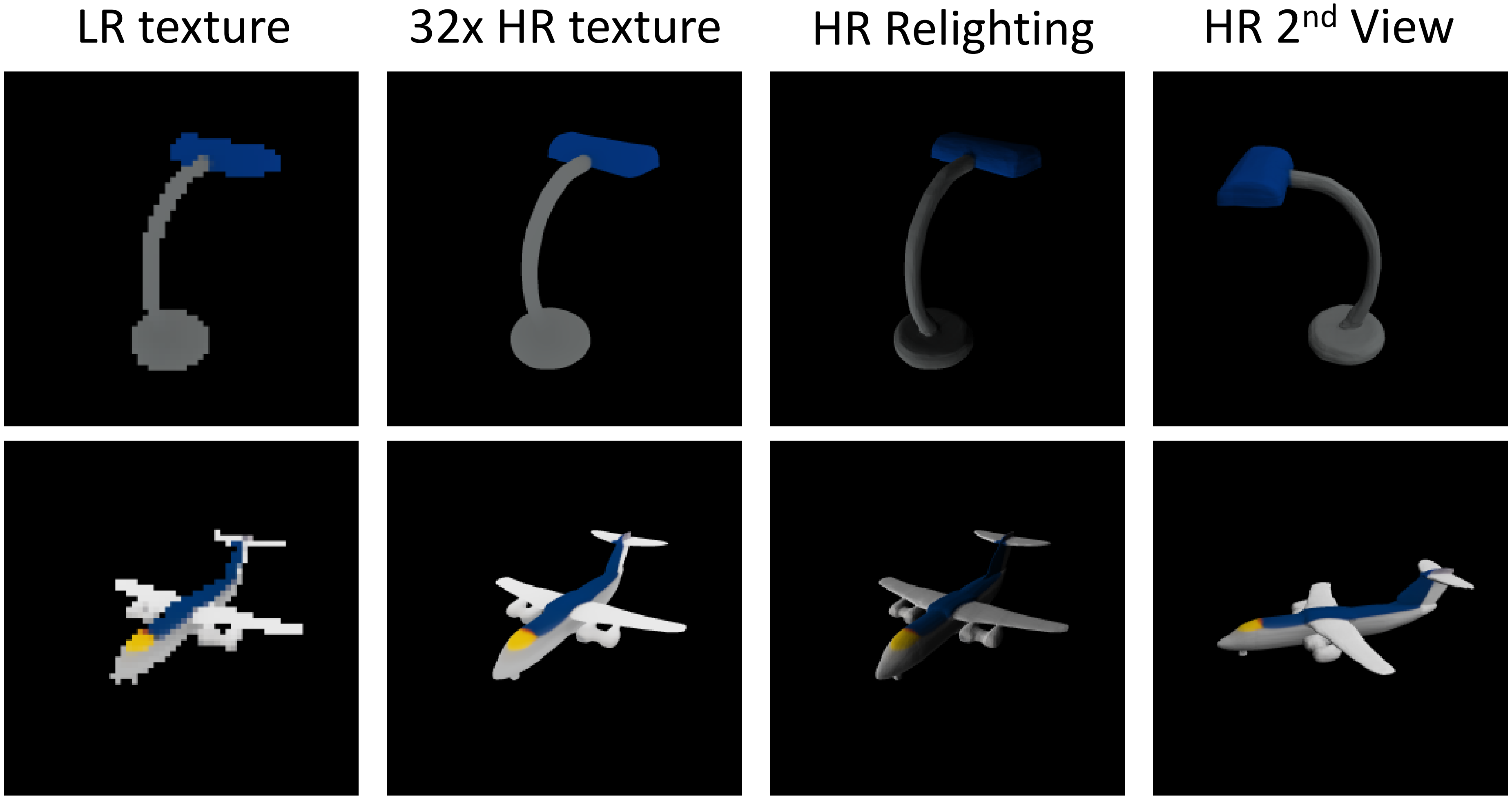
Voxel-based Representation
[Tatarchenko *et al.*]

DeepSDF Representation
[Park *et al.*]

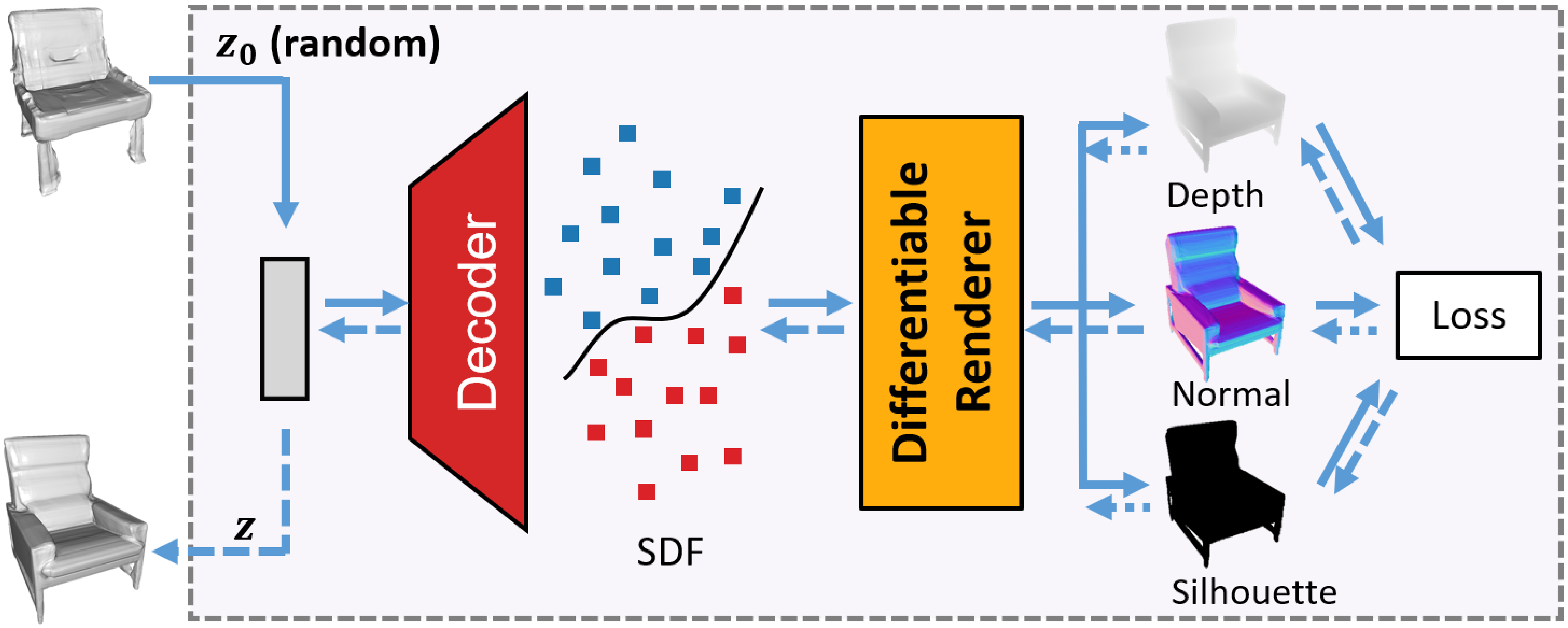
☹ **No differentiable renderer for DeepSDF!**

Park *et al.* "DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation", In CVPR 2019.

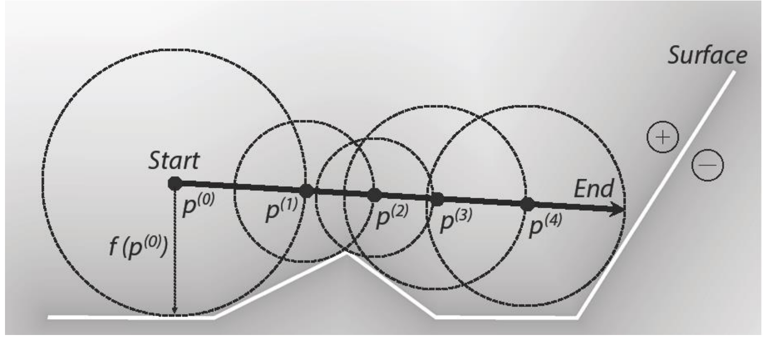
Feedforward Rendering Results



Optimization over Shape Code



DIST - Differentiable Sphere Tracing

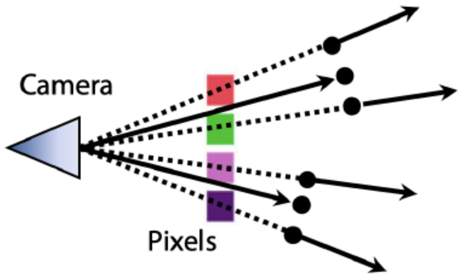


How to make feedforward efficient?

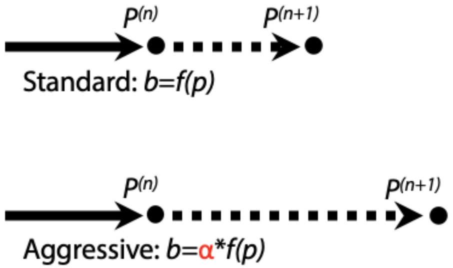
What ray convergence criteria is the best setup?

Efficient feedforward of sphere tracing algorithm

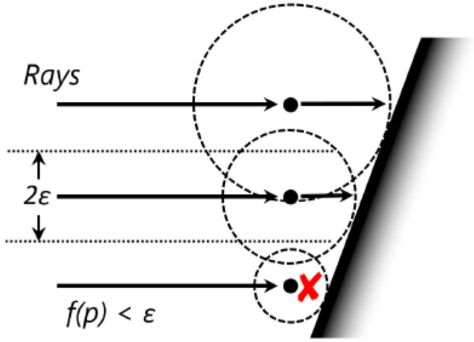
Gradient computation on the rendered silhouette



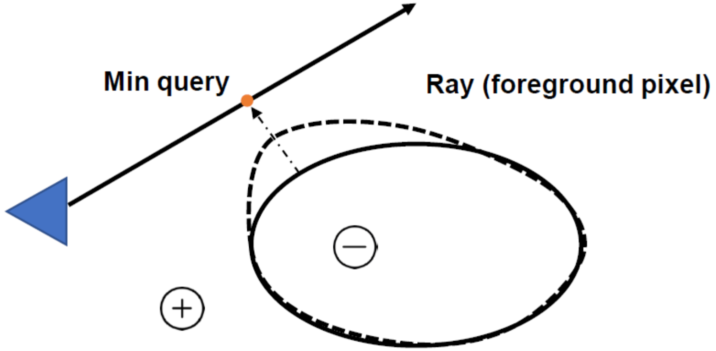
(a) Coarse-to-fine Strategy



(b) Aggressive Marching



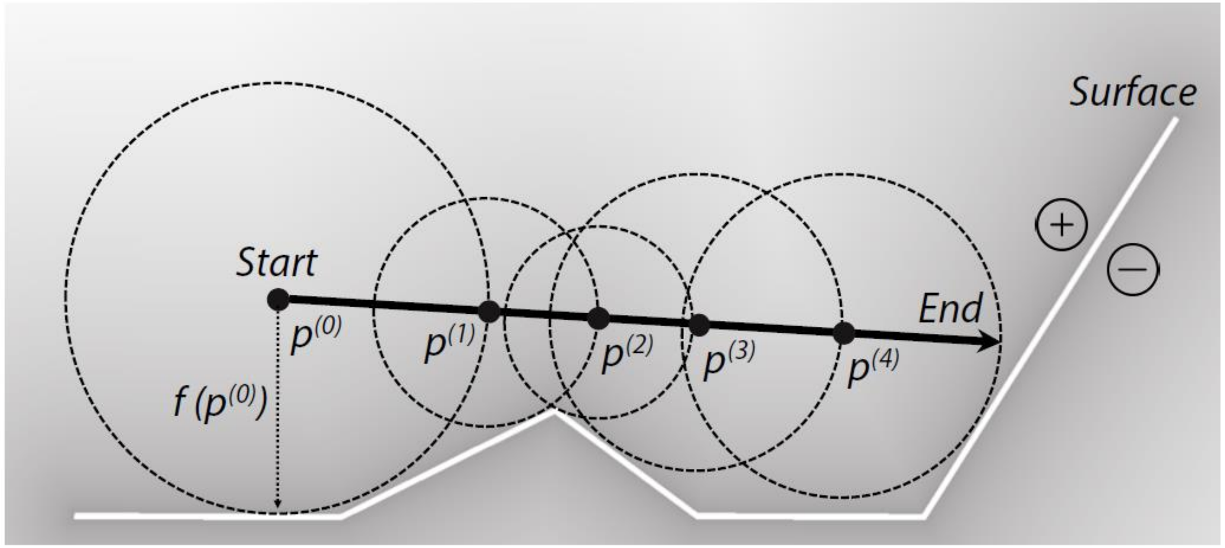
(c) Convergence Criteria



What can we do to resolve the GPU memory burden?

How to deal with non-differentiable rendered silhouette?

DIST Feedforward – Naive Sphere Tracing

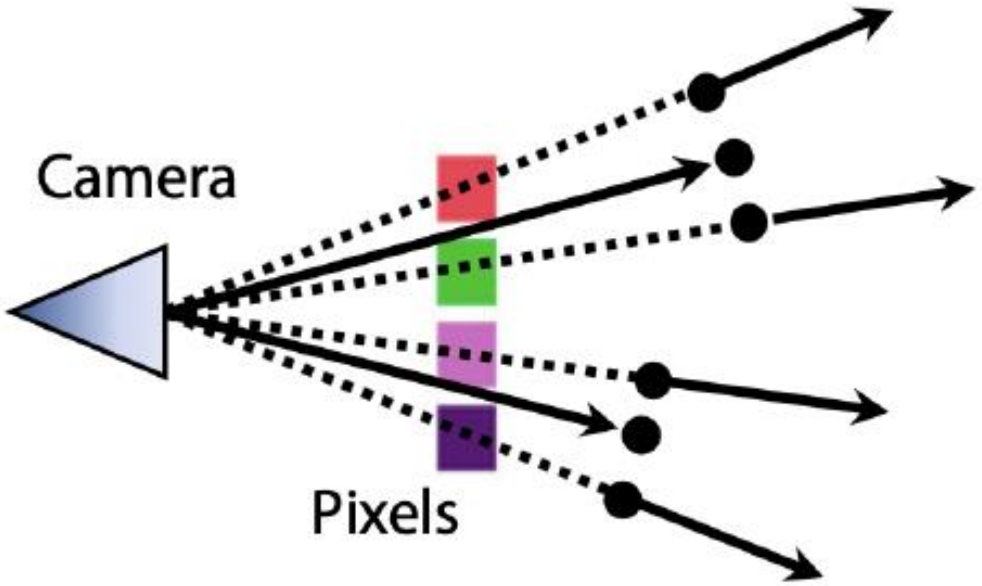


Algorithm 1 Naive sphere tracing algorithm for a camera ray $L : \mathbf{c} + d\tilde{\mathbf{v}}$ over a signed distance fields $f : \mathbb{N}^3 \rightarrow \mathbb{R}$.

Initialize $n = 0$, $d^{(0)} = 0$, $\mathbf{p}^{(0)} = \mathbf{c}$.
while not converged **do**:
  Take the corresponding SDF value $b^{(n)} = f(\mathbf{p}^{(n)})$ of the location $\mathbf{p}^{(n)}$ and make update: $d^{(n+1)} = d^{(n)} + b^{(n)}$.
  $\mathbf{p}^{(n+1)} = \mathbf{c} + d^{(n+1)}\tilde{\mathbf{v}}$, $n = n + 1$.
  Check convergence.
end while

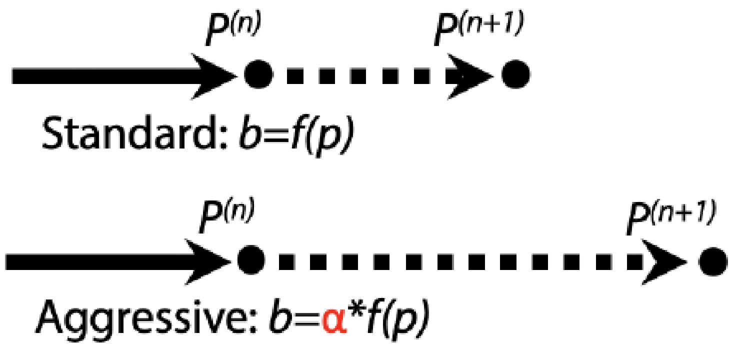
For each camera ray, march along the ray direction at each step with the queried SDF value until convergence.

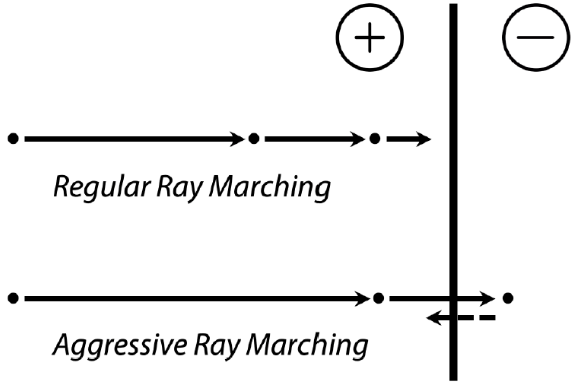
DIST Feedforward - Coarse-to-Fine Strategy



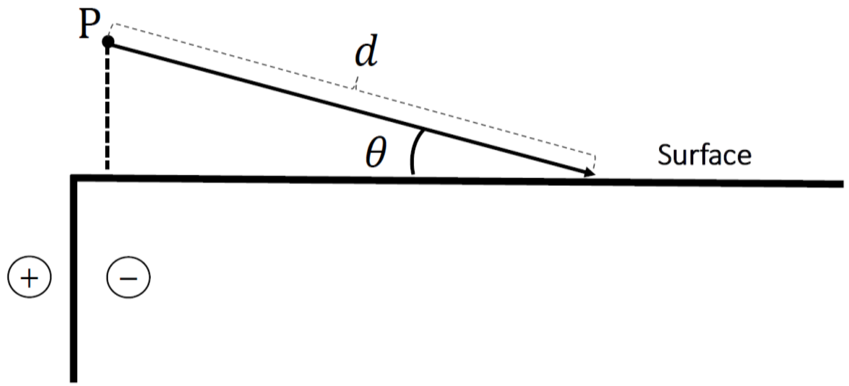
We start the sphere tracing over an image with ¼ resolution, and split each ray twice during the marching process, which saves computation at the early stage.

DIST Feedforward – Aggressive Marching





Setting step size α > 1 incurs bouncing between both sides.

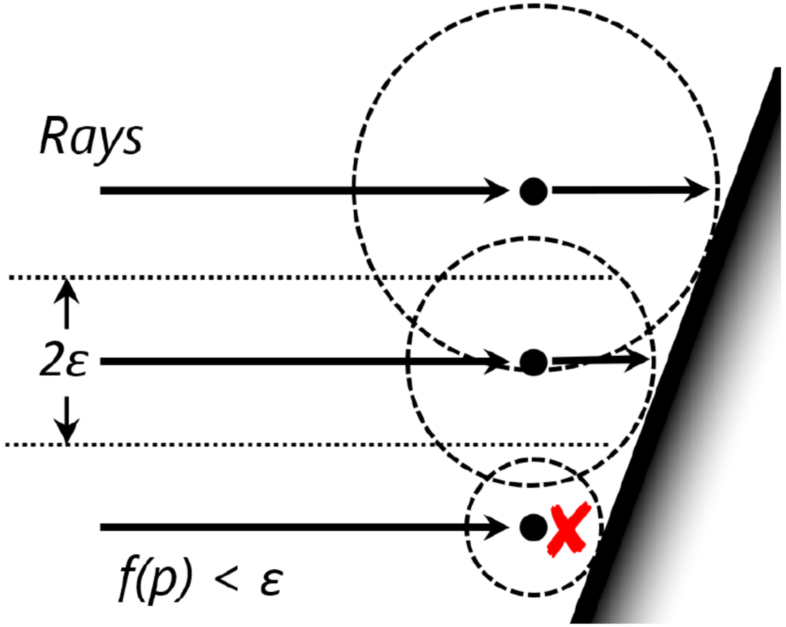


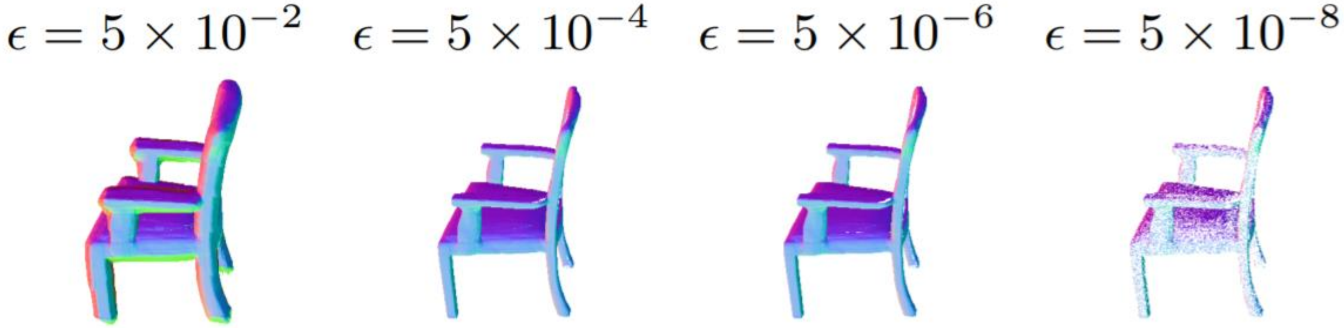
$$|d(1 - \alpha sin\theta)^k| < \epsilon$$

$$k > k_{min} = \frac{log\epsilon - logd}{log|1 - \alpha sin\theta|}$$

Setting step size α > 1 speeds up convergence.

DIST Feedforward – Convergence Criteria

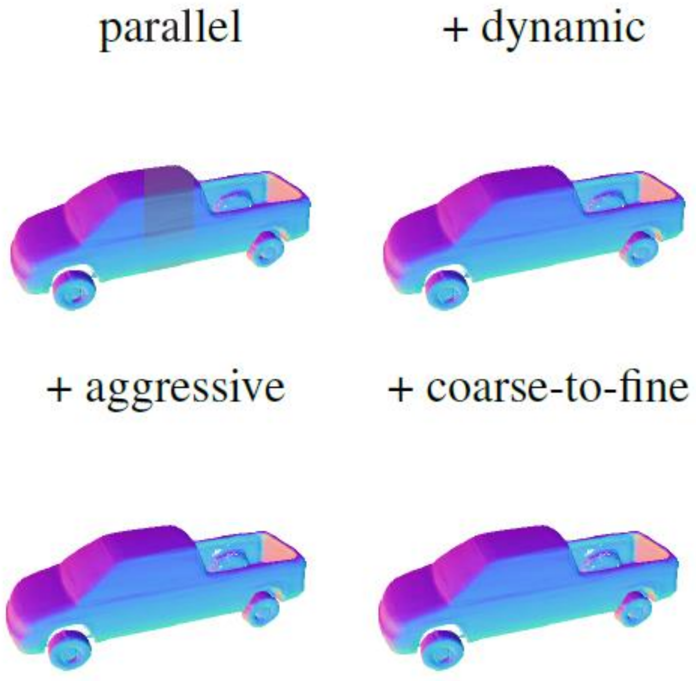




We stop the marching once the SDF value is smaller than 1/2 ϵ.

A large threshold ϵ causes dilation, while a small threshold leads to erosion.

DIST Feedforward – Results



Method	size	#step	#query	time
Naive sphere tracing	512^2	50	N/A	N/A
+ practical grad.	512^2	50	6.06M	1.6h
+ parallel	512^2	50	6.06M	3.39s
+ dynamic	512^2	50	1.99M	1.23s
+ aggressive	512^2	50	1.43M	1.08s
+ coarse-to-fine	512^2	50	**887K**	**0.99s**
+ coarse-to-fine	512^2	100	**898K**	**1.24s**

The computation becomes affordable while the results remain almost unchanged.

DIST Backward – Recursive Gradients

Each query location depends on the previous ones, incurring recursive gradients.

$$d = \alpha \sum_{n=0}^{N-1} f(\mathbf{p}^{(n)}) + (1-\alpha) f(\mathbf{p}^{(N-1)}) = d' + e$$

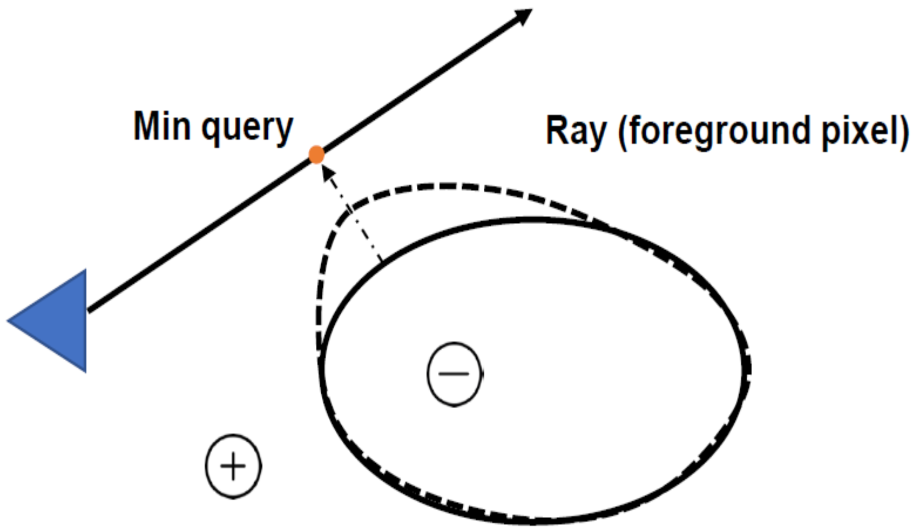
$$\frac{\partial d'}{\partial \mathbf{z}}\Big|_{\mathbf{z}_0} = \alpha \sum_{i=0}^{N-1} \frac{\partial f_\theta(\mathbf{p}^{(i)}(\mathbf{z}), \mathbf{z})}{\partial \mathbf{z}}\Big|_{\mathbf{z}_0}$$

$$= \alpha \sum_{i=0}^{N-1} \Big(\frac{\partial f_\theta(\mathbf{p}^{(i)}(\mathbf{z}_0), \mathbf{z})}{\partial \mathbf{z}} + \frac{\partial f_\theta(\mathbf{p}^{(i)}(\mathbf{z}), \mathbf{z}_0)}{\partial \mathbf{p}^{(i)}(\mathbf{z})} \frac{\partial \mathbf{p}^{(i)}(\mathbf{z}_0)}{\partial \mathbf{z}} \Big)$$

Omitted

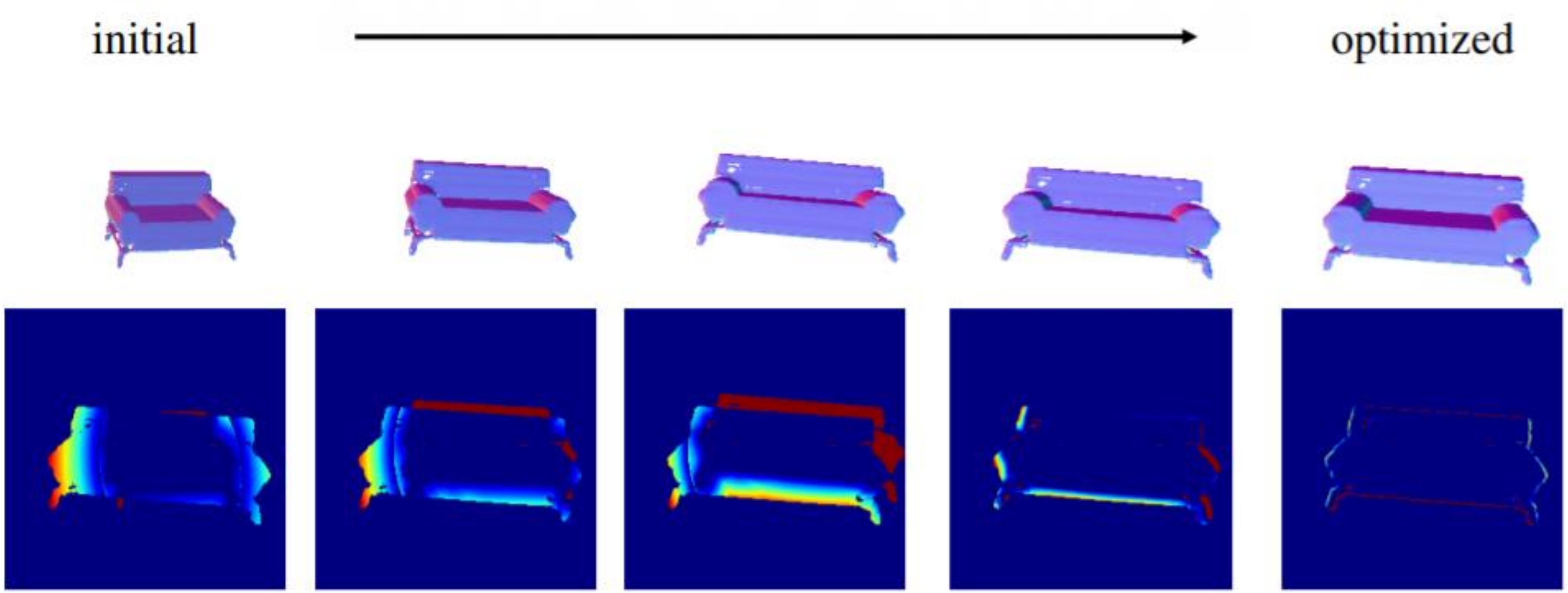
This term is omitted as it empirically has less impact on the optimization process.

DIST Backward – Differentiable Silhouette



We make use of the nice property of signed distance function to optimize the nearest surface geometry.

Optimization over Camera Parameters



Given a fixed shape, our differentiable renderer can successfully backpropagate gradients to the camera parameters with respect to 2D observations.

Results - Reconstruction from Sparse Depth Images

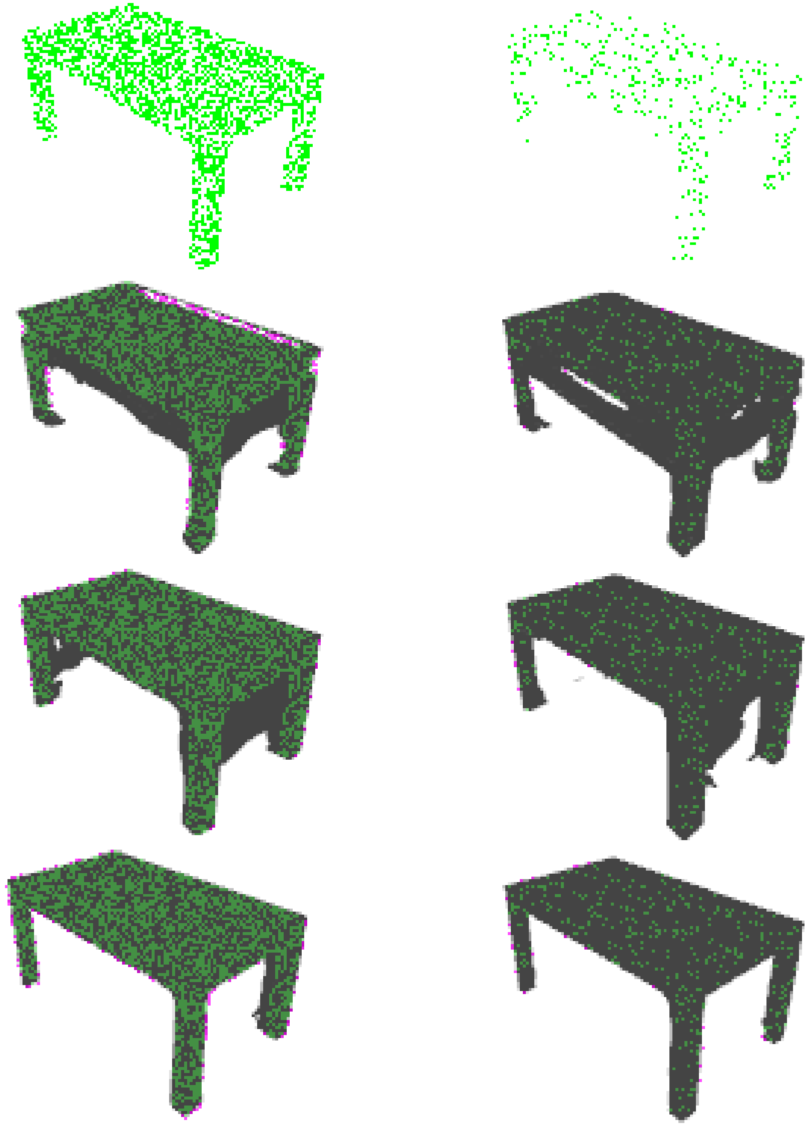
50% 10%

Input

DeepSDF

Ours (w/o mask)

Ours (w/ mask)



	dense	50%	10%	100pts	50pts	20pts
sofa						
DeepSDF	5.37	5.56	5.50	5.93	6.03	7.63
Ours	**4.12**	5.75	5.49	5.72	5.57	6.95
Ours (mask)	**4.12**	**3.98**	**4.31**	**3.98**	**4.30**	**4.94**
plane						
DeepSDF	3.71	3.73	4.29	4.44	4.40	5.39
Ours	**2.18**	4.08	4.81	4.44	4.51	5.30
Ours (mask)	**2.18**	**2.08**	**2.62**	**2.26**	**2.55**	**3.60**
table						
DeepSDF	12.93	12.78	11.67	12.87	13.76	15.77
Ours	**5.37**	12.05	11.42	11.70	13.76	15.83
Ours (mask)	**5.37**	**5.15**	**5.16**	**5.26**	**6.33**	**7.62**

Results - Reconstruction from Video Sequences

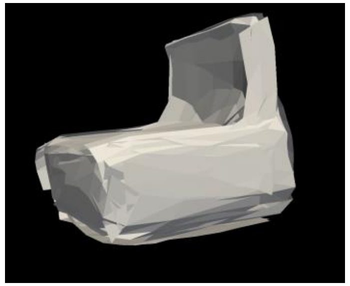
Synthetic #1

Lin *et al.*

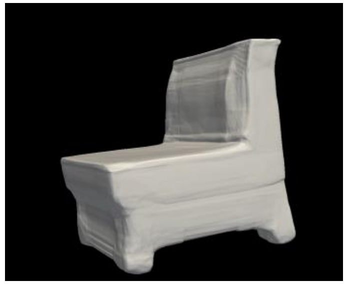
Ours

Real-world #1

Lin *et al.*



Ours



Synthetic #2

Lin *et al.*

Ours

Real-world #2

Lin *et al.*

Ours

Lin *et al*, "Photometric Mesh Optimization for Video-Aligned 3D Object Reconstruction", In CVPR 2019.

CVPR SEATTLE WASHINGTON JUNE 16-18 2020

Code and Demo are available here

↓

http://b1ueber2y.me/projects/DIST-Renderer/

ETH zürich · Google · Microsoft

DIST: Rendering Deep Implicit Signed Distance Function with Differentiable Sphere Tracing

CVPR SEATTLE WASHINGTON JUNE 16-18 2020

Shaohui Liu, Yinda Zhang, Songyou Peng, Boxin Shi, Marc Pollefeys, Zhaopeng Cui

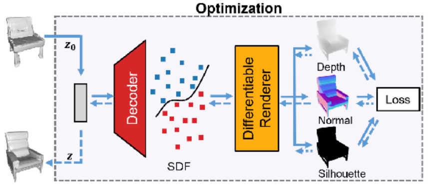
ETH zürich Google Microsoft

Motivation & Pipeline

Project Page

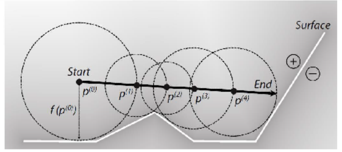
The recently proposed deep implicit signed distance function [1] is effective on representing 3D shapes. Advantages: infinite resolution, lightweight, etc.

☹ **No differentiable renderer exists** for this representation, making it infeasible to be optimized over 2D observations.



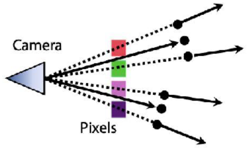
DIST – Feedforward

Naive Sphere Tracing



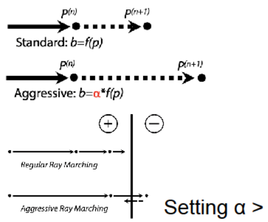
For each camera ray, march at each step with the queried SDF value until convergence.

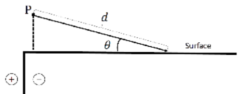
Coarse-to-fine Strategy



We start the sphere tracing over an image with ¼ resolution, and split each ray twice during the marching process, which saves computation at the early stage.

Aggressive Marching



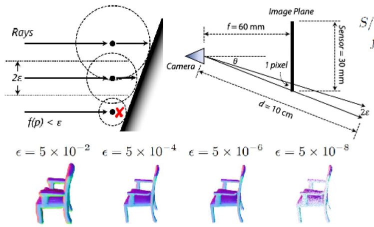


$$|d(1-\alpha sin\theta)^k| < \epsilon$$

$$k > k_{min} = \frac{log\epsilon - logd}{log|1-\alpha sin\theta|}$$

Setting α > 1 speeds up convergence.

Convergence Criteria



$$\frac{S/R \cdot cos(\theta)}{f/cos(\theta)} = \frac{2\epsilon}{d_{min}} \quad \epsilon = \frac{d_{min}\cdot S \cdot cos^2(\theta)}{2\cdot f\cdot R}$$

Take focal length	f = 60mm,
sensor size	S = 32mm,
resolution	R = 512,
minimum depth	d_{min} = 10cm,
We can get	$\epsilon = 5 \times 10^{-5}$.

A large threshold causes dilation, while a small threshold leads to erosion.

DIST - Backward

Memory issue caused by Recursive Gradients

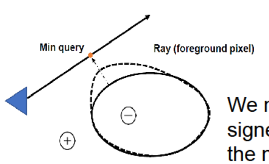
$$d = \alpha\sum_{n=0}^{N-1} f(\mathbf{p}^{(n)}) + (1-\alpha)f(\mathbf{p}^{(N-1)}) = d' + e$$

$$\frac{\partial d'}{\partial \mathbf{z}}|_{\mathbf{z}_0} = \alpha\sum_{i=0}^{N-1}\frac{\partial f_\theta(\mathbf{p}^{(i)}(\mathbf{z}),\mathbf{z})}{\partial \mathbf{z}}|_{\mathbf{z}_0}$$

$$= \alpha\sum_{i=0}^{N-1}\left(\frac{\partial f_\theta(\mathbf{p}^{(i)}(\mathbf{z}_0),\mathbf{z})}{\partial \mathbf{z}} + \frac{\partial f_\theta(\mathbf{p}^{(i)}(\mathbf{z}),\mathbf{z}_0)}{\partial \mathbf{p}^{(i)}(\mathbf{z})}\frac{\partial \mathbf{p}^{(i)}(\mathbf{z}_0)}{\partial \mathbf{z}}\right)$$

Each query location depends on the previous one, incurring recursive gradients. We make approximations over sphere tracing by omitting high-order gradients.

Differentiable Silhouette



We make use of the nice property of signed distance function to optimize the nearest surface geometry.

Feedforward Rendering

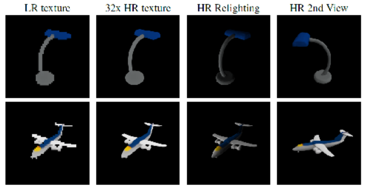
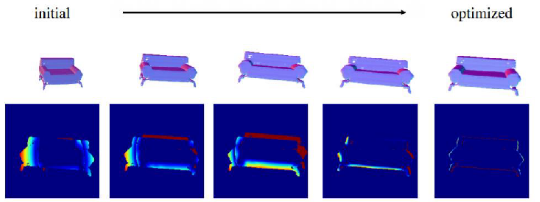


Image size = 512 x 512
marching step = 50

Method	#query	time
Naive sphere tracing	N/A	N/A
+ practical grad.	6.06M	1.6h
+ parallel	6.06M	3.39s
+ dynamic	1.99M	1.23s
+ aggressive	1.43M	1.08s
+ coarse-to-fine	**887K**	**0.99s**

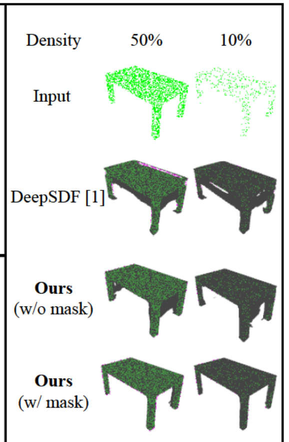
Optimization over Camera Parameters



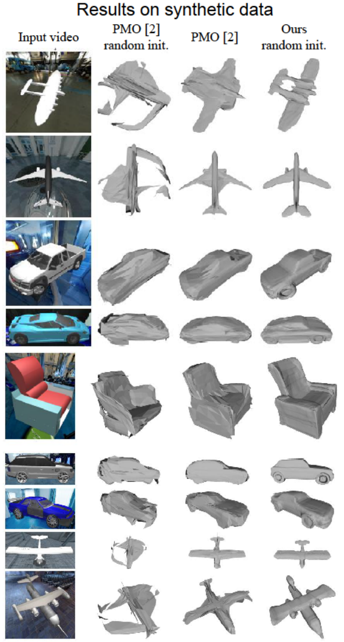
Reconstruction from Sparse Depths

Quantitative evaluation

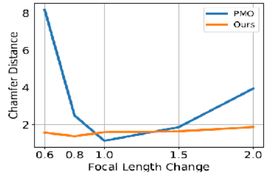
	dense	50%	10%	100pts	50pts	20pts
sofa						
DeepSDF	5.37	5.56	5.50	5.93	6.03	7.63
Ours	**4.12**	5.75	5.49	5.72	5.57	6.95
Ours (mask)	**4.12**	**3.98**	**4.31**	**3.98**	**4.30**	**4.94**
plane						
DeepSDF	3.71	3.73	4.29	4.44	4.40	5.39
Ours	**2.18**	4.08	4.81	4.44	4.51	5.30
Ours (mask)	**2.18**	**2.08**	**2.62**	**2.26**	**2.55**	**3.60**
table						
DeepSDF	12.93	12.78	11.67	12.87	13.76	15.77
Ours	**5.37**	12.05	11.42	11.70	13.76	15.83
Ours (mask)	**5.37**	**5.15**	**5.16**	**5.26**	**6.33**	**7.62**



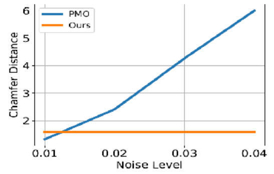
Reconstruction from Video Sequences



Generalization across different focal lengths



Generalization across different noise levels



References:
[1] Park et al. "DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation", CVPR'19.
[2] Lin et al, "Photometric Mesh Optimization for Video-Aligned 3D Object Reconstruction", CVPR '19.

Full-Resolution: http://b1ueber2y.me/projects/DIST-Renderer/pdf/4986-poster.pdf