Volume Rendering of Neural Implicit Surfaces

Lior Yariv   Jiatao Gu   Yoni Kasten   Yaron Lipman
Weizmann Institute of Science
Facebook AI Research

TABLE OF CONTENTS

01

INTRODUCTION

Background and Related Works

INTRODUCTION

- Volume Rendering
- Neural Radiance Field (NeRF)
    - In nutshell
    - Drawbacks?
- VolSDF Contribution
    - New density representation and reconstruction
        - Moving from generic to geometric density function

RELATED WORKS

- Neural Scene Representation & Rendering
  - Combining neural implicit functions with volume rendering
    - \+ Expressive representation power
    - \+ low memory foot-print
    - − Recovered geometry
    - − Opacity function approximation

RELATED WORKS

- Multi-view 3D Reconstruction
  - Depth-based approaches
    - – Complex pipeline
  - Voxel-based approaches
    - \+ Directly model objects in a volume
    - – Limited to low resolution
    - – Require accurate object masks

03
METHODS

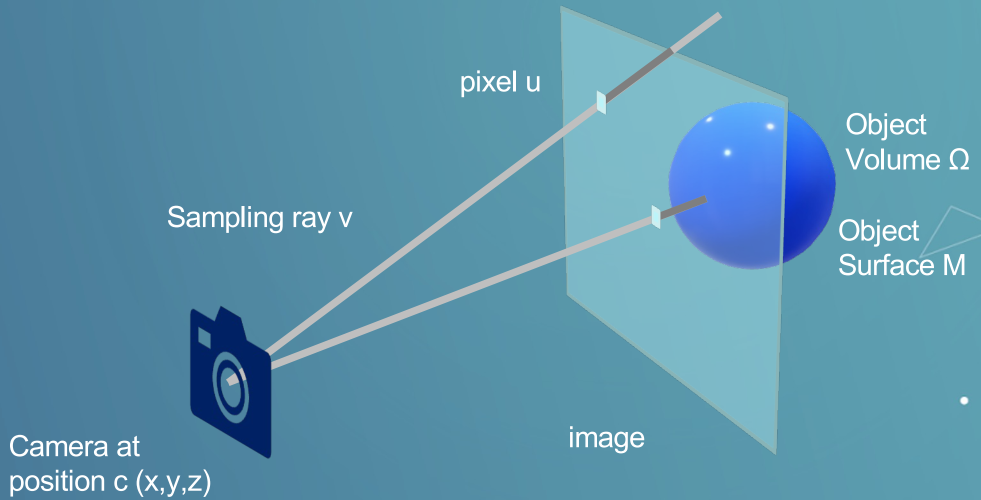
A New Geometric Model Density

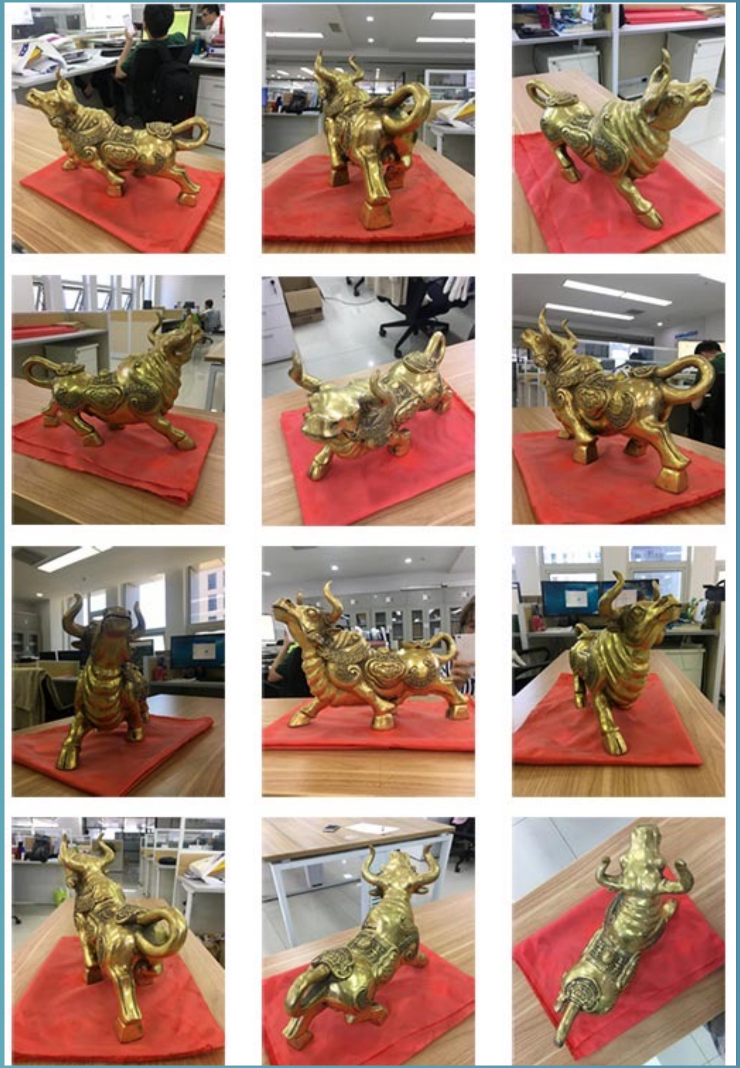
(1/5) Density As Transformed SDF

- What is SDF?
- Drawback of current model
- Improvement of previous model
- Advantages

(1/5) Density As Transformed SDF

- What is SDF?
  - A signed distance function is an n-dimensional implicit function, which associates a scalar value with each point of its n-dimensional domain.





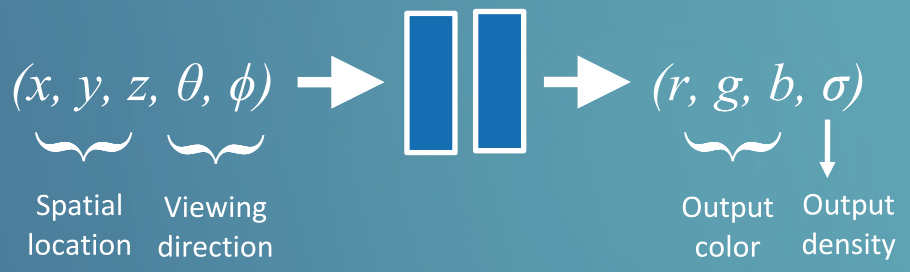
volumetric density

signed distance function

produced rendering

(1/5) Density As Transformed SDF

- What is SDF?
- Drawback of current model



(1/5) Density As Transformed SDF

- What is SDF?
- Drawback of current model

$(x, y, z, \theta, \phi) \rightarrow$  $\rightarrow (r, g, b, \sigma)$

- Geometric volume density at point x

$$\sigma(x) = \alpha \Psi_\beta(-d_\Omega(x)) \quad (2)$$

$$d_\Omega(x) = (-1)^{1_\Omega(x)} \min_{y \in \mathcal{M}} \|x - y\| \quad (1)$$

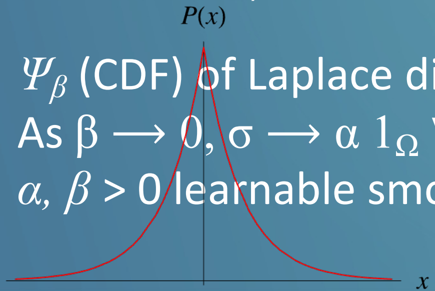
indicator if $x \in$ volume Ω

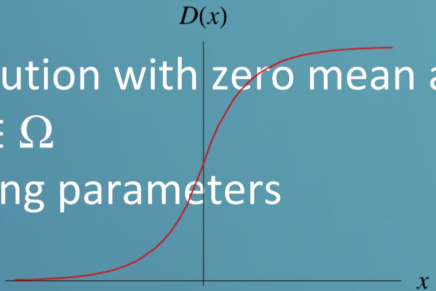
point y at surface M

Ψ_β (CDF) of Laplace distribution with zero mean and β scale

As $\beta \longrightarrow 0, \sigma \longrightarrow \alpha\, 1_\Omega\ \forall\, x \in \Omega$

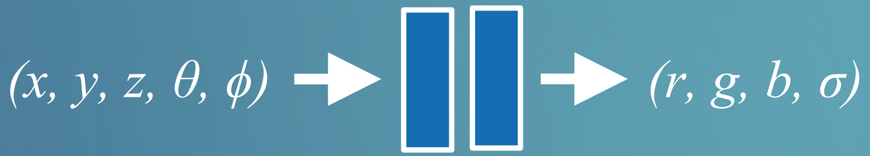
$\alpha, \beta > 0$ learnable smoothing parameters





(1/5) Density As Transformed SDF

- What is SDF?
- Drawback of current model



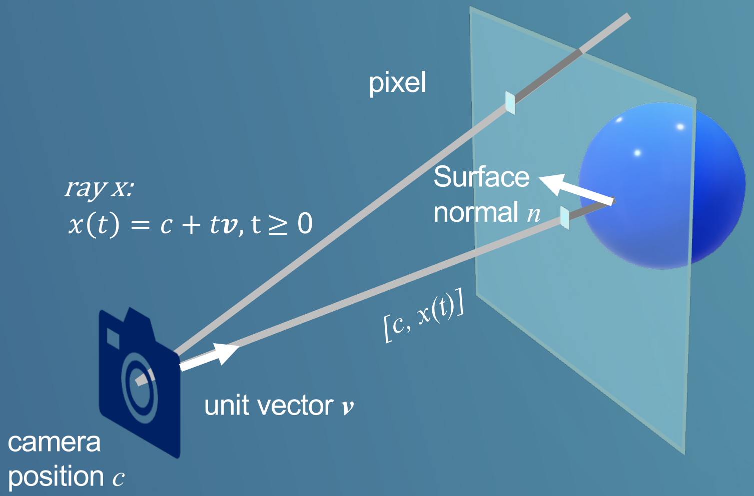
- Geometric volume density

$$\sigma(x) = \alpha \Psi_\beta \left(-d_\Omega(x)\right) \quad (2)$$

- Advantages

(2/5) Volume Rendering of σ

- Volume rendering integral
  - approximating the integrated (i.e., summed) light radiance along this ray reaching the camera



The probability a light particle succeeds traversing the segment [c, x(t)] without bouncing off

$$T(t) = \exp(-\int_0^t \sigma(x(s))ds) \quad (4)$$

The volume rendering equation is the expected light along the ray,

$$I(c, v) = \int_0^\infty L(x(t), n(t), v)\tau(t)dt \quad (7)$$

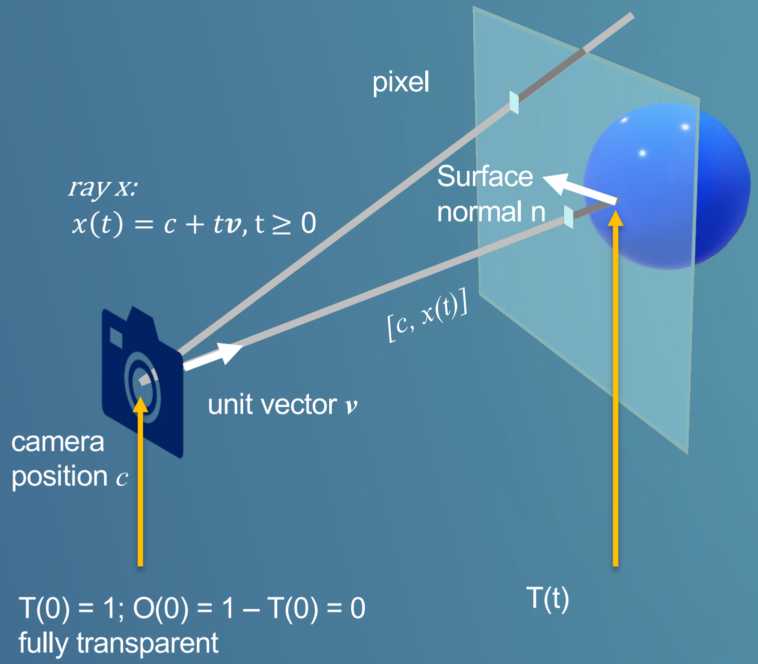
where $L(x, n, v)$ is the radiance field

(2/5) Volume Rendering of σ

- Opacity as CDF:

$$T(t) = \exp(-\int_0^t \sigma(x(s))ds) \quad (4)$$

$$\tau(t) = \sigma(x(t))\ T(t) = \frac{dO}{dt}(t) \quad PDF$$

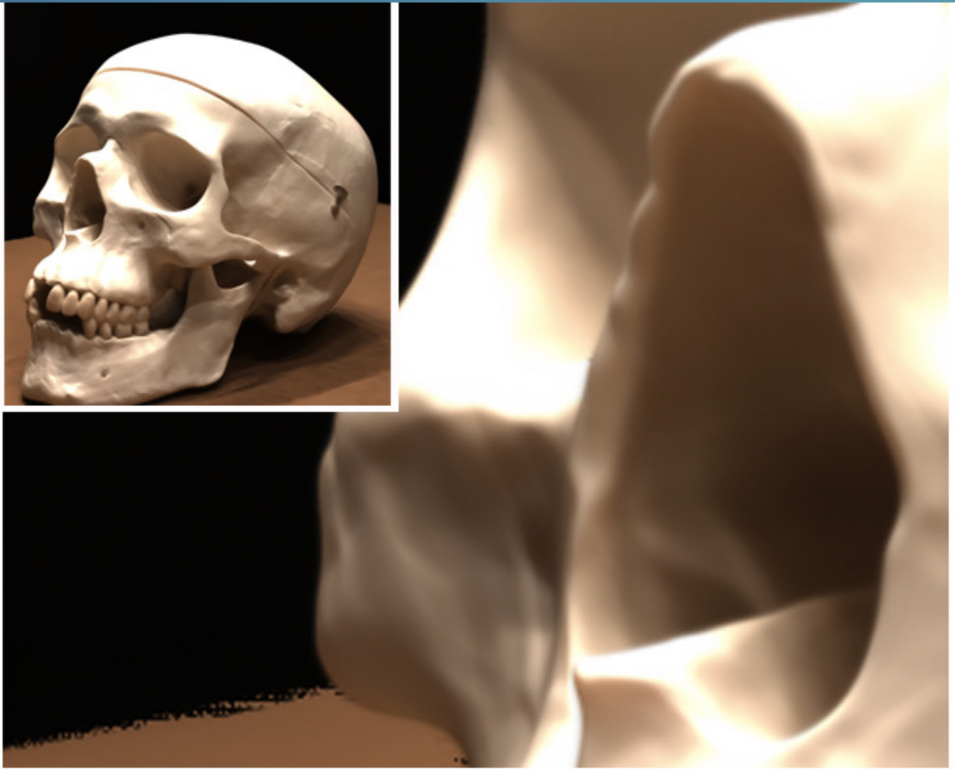


(2/5) Volume Rendering of σ

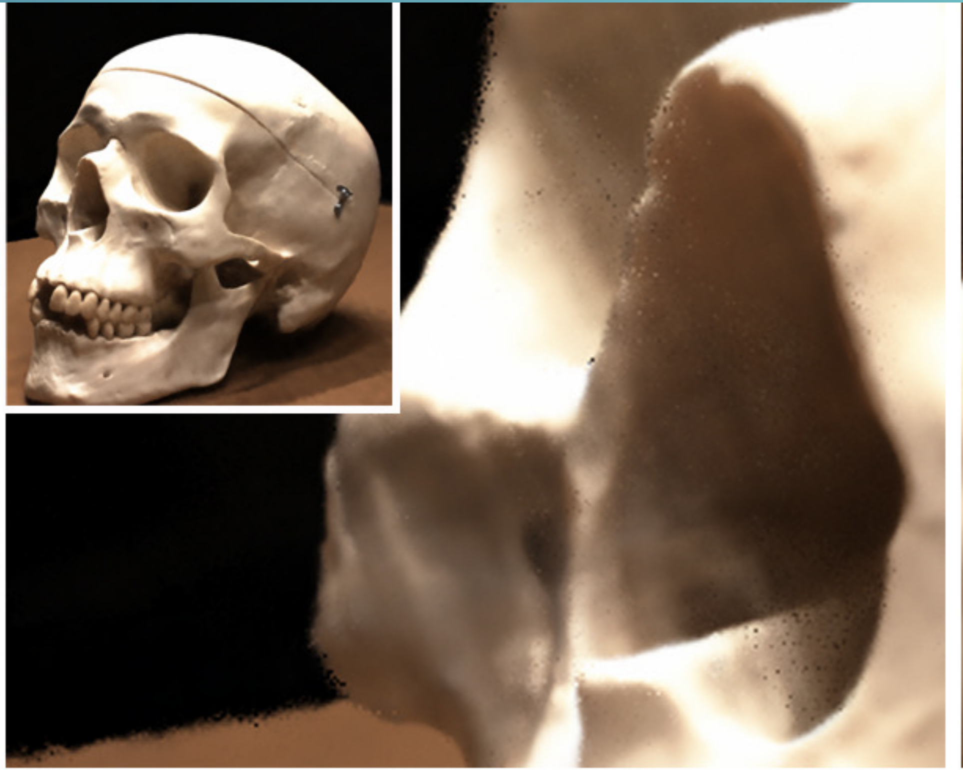
- Sampling τ
  - PDF τ is typically extremely concentrated near the object's boundary
  - the choice of the sample points S has a crucial effect on the approximation quality
  - NeRF: second, coarse network was trained specifically for the approximation of the opacity
  - VolSDF: sampling S is computed by a sampling algorithm based on an error bound for the opacity approximation

(2/5) Volume Rendering of σ

VolSFD

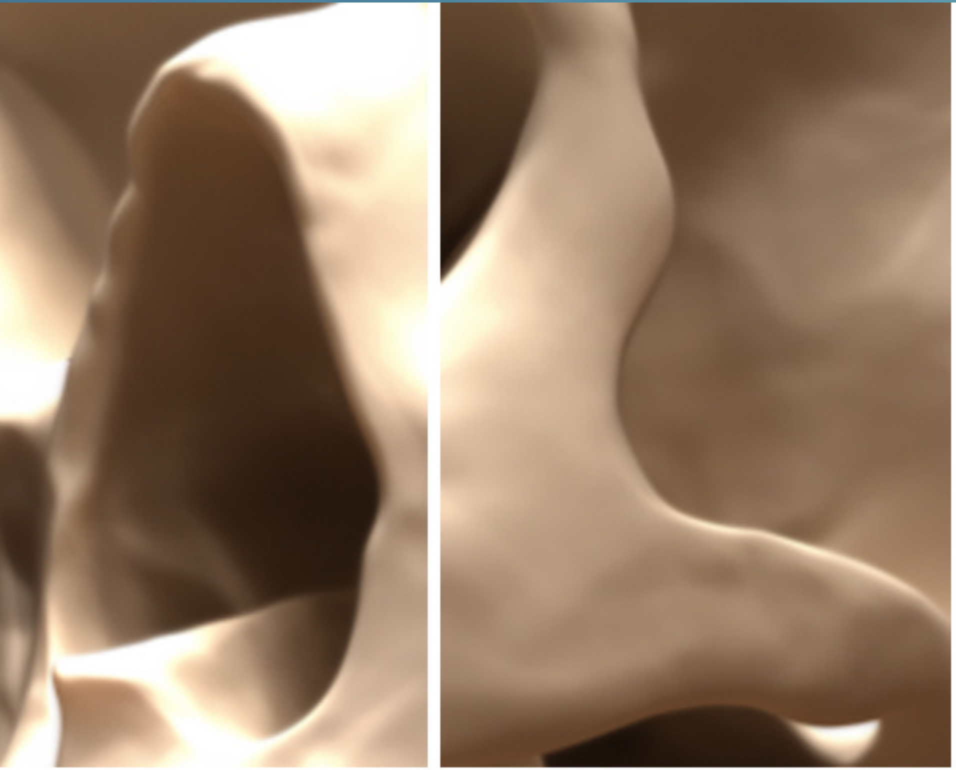


NeRF

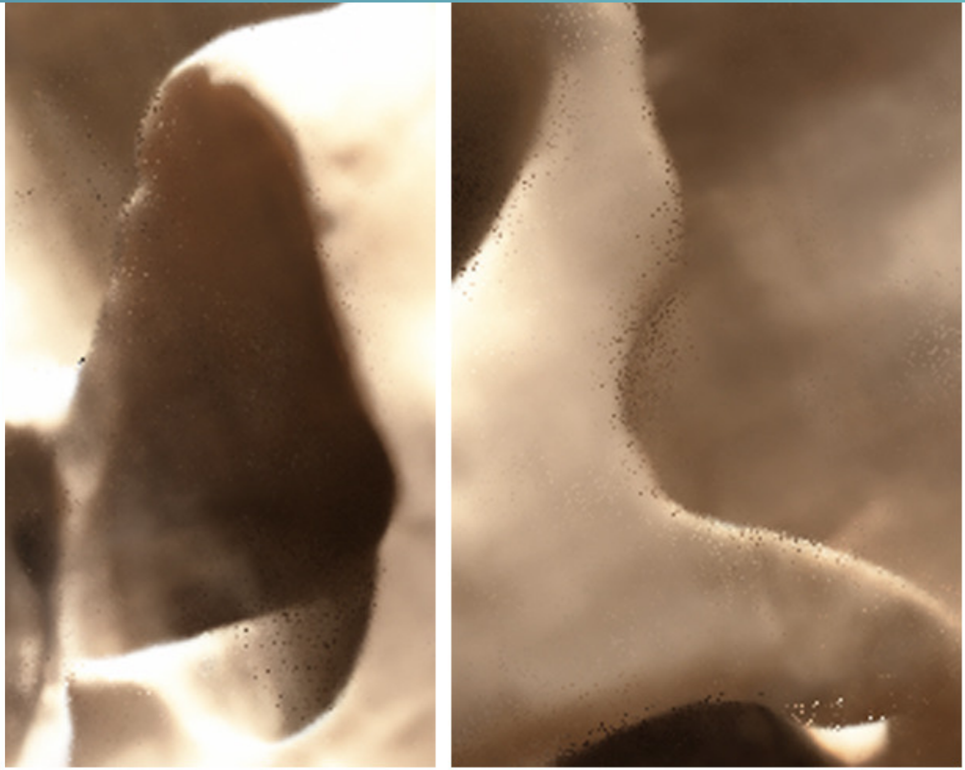


(2/5) Volume Rendering of σ

VolSFD



NeRF



(3/5) Bound on The Opacity Approximation Error

- Transparency for ray x with sample point s:

$$T(t) = \exp(-\int_0^t \sigma(x(s))ds)$$

$$O(t) = 1 - T(t)$$

$$O(t) = 1 - \exp(-\int_0^t \sigma(x(s))ds)$$

$$O(t) = 1 - \exp(-\hat{R}(t))$$

$$\hat{R}(t) = \sum_{i=1}^{k-1} \delta_i \sigma_i + (t - t_k)\sigma_k$$

(3/5) Bound on The Opacity Approximation Error

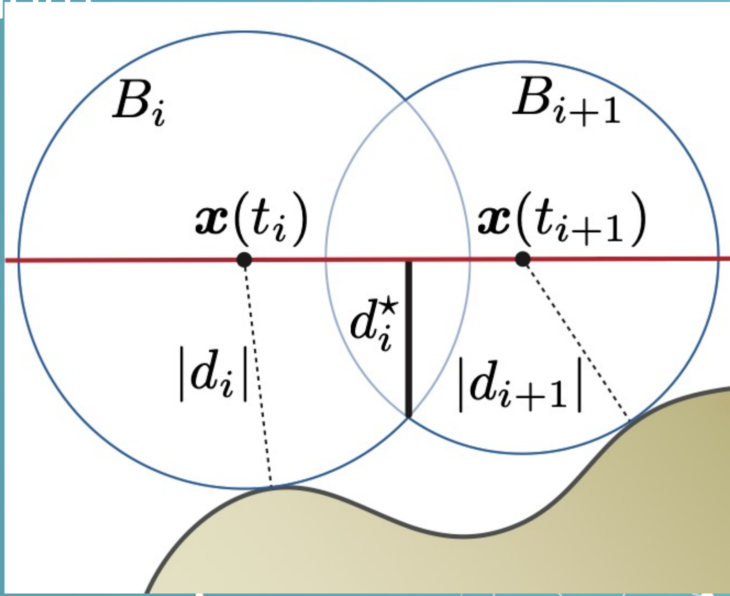
- What about the error in summation?

$$\sigma(x) = \alpha\Psi_\beta(-d_\Omega(x))$$

$$\Psi_\beta = \begin{cases} \frac{1}{2}\exp\left(\frac{s}{\beta}\right), s \leq 0 \\ 1 - \frac{1}{2}\exp\left(-\frac{s}{\beta}\right), s > 0 \end{cases}$$

$$\frac{d}{dt}\Psi_\beta = \frac{1}{2\beta}\exp\left(-\frac{|s|}{\beta}\right)$$

$$\left|\frac{d}{ds}\boldsymbol{\sigma}(x(s))\right| \leq \frac{\alpha}{2\beta}\exp\left(-\frac{d_i^*}{\beta}\right), where\ d_i^* = \min_{\substack{s\in[t_i,t_{i+1}] \\ y\notin B_i\cup B_{i+1}}} \|x(s) - y\|$$



(3/5) Bound on The Opacity Approximation Error

- This bound can be used to derive an error bound for $\hat{R}$

$$\left|\frac{d}{ds}\sigma(x(s))\right| \leq \frac{\alpha}{2\beta}\exp\left(-\frac{d_i^*}{\beta}\right), where\ d_i^* = \min_{\substack{s\in[t_i,t_{i+1}] \\ y\notin B_i \cup B_{i+1}}} \|x(s)-y\|$$

$$\hat{R}(t) = \sum_{i=1}^{k-1} \delta_i \sigma_i + (t - t_k)\sigma_k$$

$$|E(t)| \leq \hat{E}(t) = \frac{\alpha}{4\beta}\left(\sum_{i=1}^{k-1} \delta_i^2 \exp\left(-\frac{d_i^*}{\beta}\right) + (t-t_k)^2 \exp\left(-\frac{d_k^*}{\beta}\right)\right).$$

- So the error of the appr. opacity Ô can be bounded as

$$\left|O(t) - \hat{O}(t)\right| \leq \exp\left(-\hat{R}(t)\right)\left(\exp\left(\hat{E}(t)\right) - 1\right)$$

(3/5) Bound on The Opacity Approximation Error

- Taking the maximum over all intervals furnishes a bound $B_{\mathcal{T},\beta}$ as a function of $\mathcal{T}$ and β

$$\left|O(t) - \hat{O}(t)\right| \leq \exp\left(-\hat{R}(t)\right)\left(\exp\left(\hat{E}(t)\right) - 1\right)$$

$$\max_{t\in[o,M]}\left|O(t) - \hat{O}(t)\right| \leq B_{\mathcal{T},\beta} = \max_{k\in[n-1]}\left\{\exp\left(-\hat{R}(t_k)\right)\left(\exp\left(\hat{E}(t_{k+1})\right) - 1\right)\right\}$$

- Where is $\mathcal{T}$ a set of samples

$$\mathcal{T} = \{t_i\}_{i=1}^{n}, 0 = t_1 < \cdots < t_n = M$$

(4/5) Sampling Algorithm

- Using the bound to compute sampling:
- $I(\boldsymbol{c}, \boldsymbol{v}) = \int_0^\infty L(x(t), n(t), v)\tau(t)dt$
- $I(\boldsymbol{c}, \boldsymbol{v}) \approx \hat{I}_S(\boldsymbol{c}, \boldsymbol{v}) = \sum_{i=1}^{m-1} \hat{\tau}_i L_i$

Algorithm 1: Sampling algorithm.

Input: error threshold $\epsilon > 0$; β

Initialize $\mathcal{T} = \mathcal{T}_0$

Initialize β_+ such that $B_{\mathcal{T},\beta_+} \leq \epsilon$

while $B_{\mathcal{T},\beta} > \epsilon$ *and not max_iter* **do**

  upsample $\mathcal{T}$

  if $B_{\mathcal{T},\beta_+} < \epsilon$ **then**

    Find $\beta_\star \in (\beta, \beta_+)$ so that $B_{\mathcal{T},\beta_\star} = \epsilon$

    Update $\beta_+ \leftarrow \beta_\star$

  end

end

Estimate $\widehat{O}$ using $\mathcal{T}$ and β_+

$\mathcal{S} \leftarrow$ get fresh m samples using $\hat{O}^{-1}$

return $\mathcal{S}$

(4/5) Sampling Algorithm

Setting β as:

$$\beta \geq \frac{\alpha M}{4(n-1)\log(1+\epsilon)}$$

For $n > 0$, $\epsilon > 0$ and $B_{\mathcal{T},\beta} \leq \epsilon$

Here n = 128 was used.

Algorithm 1: Sampling algorithm.

Input: error threshold $\epsilon > 0$; β

Initialize $\mathcal{T} = \mathcal{T}_0$

Initialize β_+ such that $B_{\mathcal{T},\beta_+} \leq \epsilon$

while $B_{\mathcal{T},\beta} > \epsilon$ *and not max_iter* **do**

    upsample $\mathcal{T}$

    if $B_{\mathcal{T},\beta_+} < \epsilon$ **then**

        Find $\beta_\star \in (\beta, \beta_+)$ so that $B_{\mathcal{T},\beta_\star} = \epsilon$

        Update $\beta_+ \leftarrow \beta_\star$

    end

end

Estimate $\widehat{O}$ using $\mathcal{T}$ and β_+

$\mathcal{S} \leftarrow$ get fresh m samples using $\hat{O}^{-1}$

return $\mathcal{S}$

(4/5) Sampling Algorithm

We initialize $\mathcal{T}$ with uniform sampling $\mathcal{T}_0$

Algorithm 1: Sampling algorithm.

Input: error threshold $\epsilon > 0$; β

```
Initialize 𝒯 = 𝒯_0
Initialize β_+ such that B_{𝒯,β_+} ≤ ϵ
while B_{𝒯,β} > ϵ and not max_iter do
    upsample 𝒯
    if B_{𝒯,β_+} < ϵ then
        Find β_⋆ ∈ (β, β_+) so that
          B_{𝒯,β_⋆} = ϵ
        Update β_+ ← β_⋆
    end
end
Estimate Ô using 𝒯 and β_+
𝒮 ← get fresh m samples using Ô^{-1}
return 𝒮
```

(4/5) Sampling Algorithm

pick $\beta_+ > \beta$ so that the error bound satisfies the required ϵ bound

Algorithm 1: Sampling algorithm.

Input: error threshold $\epsilon > 0$; β

Initialize $\mathcal{T} = \mathcal{T}_0$

Initialize β_+ such that $B_{\mathcal{T},\beta_+} \leq \epsilon$

while $B_{\mathcal{T},\beta} > \epsilon$ *and not max_iter* **do**

  upsample $\mathcal{T}$

  if $B_{\mathcal{T},\beta_+} < \epsilon$ **then**

    Find $\beta_\star \in (\beta, \beta_+)$ so that $B_{\mathcal{T},\beta_\star} = \epsilon$

    Update $\beta_+ \leftarrow \beta_\star$

  end

end

Estimate $\widehat{O}$ using $\mathcal{T}$ and β_+

$\mathcal{S} \leftarrow$ get fresh m samples using $\hat{O}^{-1}$

return $\mathcal{S}$

(4/5) Sampling Algorithm

n samples are added to $\mathcal{T}$ to reduce β_+ while keeping $B_{\mathcal{T},\beta}$ within error bound

Algorithm 1: Sampling algorithm.

Input: error threshold $\epsilon > 0$; β

Initialize $\mathcal{T} = \mathcal{T}_0$
Initialize β_+ such that $B_{\mathcal{T},\beta_+} \leq \epsilon$
while $B_{\mathcal{T},\beta} > \epsilon$ *and not max_iter* **do**
    upsample $\mathcal{T}$
    if $B_{\mathcal{T},\beta_+} < \epsilon$ **then**
        Find $\beta_\star \in (\beta, \beta_+)$ so that $B_{\mathcal{T},\beta_\star} = \epsilon$
        Update $\beta_+ \leftarrow \beta_\star$
    end
end
Estimate $\widehat{O}$ using $\mathcal{T}$ and β_+
$\mathcal{S} \leftarrow$ get fresh m samples using $\hat{O}^{-1}$
return $\mathcal{S}$

(4/5) Sampling Algorithm

We use the bisection method (10 max iterations) to search for β_* and update β_+

Algorithm 1: Sampling algorithm.

Input: error threshold $\epsilon > 0$; β

Initialize $\mathcal{T} = \mathcal{T}_0$

Initialize β_+ such that $B_{\mathcal{T},\beta_+} \leq \epsilon$

while $B_{\mathcal{T},\beta} > \epsilon$ *and not max_iter* **do**

  upsample $\mathcal{T}$

  if $B_{\mathcal{T},\beta_+} < \epsilon$ **then**

    Find $\beta_\star \in (\beta, \beta_+)$ so that $B_{\mathcal{T},\beta_\star} = \epsilon$

    Update $\beta_+ \leftarrow \beta_\star$

  end

end

Estimate $\widehat{O}$ using $\mathcal{T}$ and β_+

$\mathcal{S} \leftarrow$ get fresh m samples using $\hat{O}^{-1}$

return $\mathcal{S}$

(4/5) Sampling Algorithm

Run iteratively until $B_{\mathcal{T},\beta} \leq \epsilon$ (5 max iter)

Algorithm 1: Sampling algorithm.

Input: error threshold $\epsilon > 0$; β

Initialize $\mathcal{T} = \mathcal{T}_0$

Initialize β_+ such that $B_{\mathcal{T},\beta_+} \leq \epsilon$

while $B_{\mathcal{T},\beta} > \epsilon$ *and not max_iter* **do**

  upsample $\mathcal{T}$

  if $B_{\mathcal{T},\beta_+} < \epsilon$ **then**

    Find $\beta_\star \in (\beta, \beta_+)$ so that $B_{\mathcal{T},\beta_\star} = \epsilon$

    Update $\beta_+ \leftarrow \beta_\star$

  end

end

Estimate $\widehat{O}$ using $\mathcal{T}$ and β_+

$\mathcal{S} \leftarrow$ get fresh m samples using $\hat{O}^{-1}$

return $\mathcal{S}$

(4/5) Sampling Algorithm

Use final $\mathcal{T}$ and β_+ to est. opacity $\hat{O}$

Algorithm 1: Sampling algorithm.

Input: error threshold $\epsilon > 0$; β

Initialize $\mathcal{T} = \mathcal{T}_0$

Initialize β_+ such that $B_{\mathcal{T},\beta_+} \leq \epsilon$

while $B_{\mathcal{T},\beta} > \epsilon$ *and not max_iter* **do**

  upsample $\mathcal{T}$

  if $B_{\mathcal{T},\beta_+} < \epsilon$ **then**

    Find $\beta_\star \in (\beta, \beta_+)$ so that $B_{\mathcal{T},\beta_\star} = \epsilon$

    Update $\beta_+ \leftarrow \beta_\star$

  end

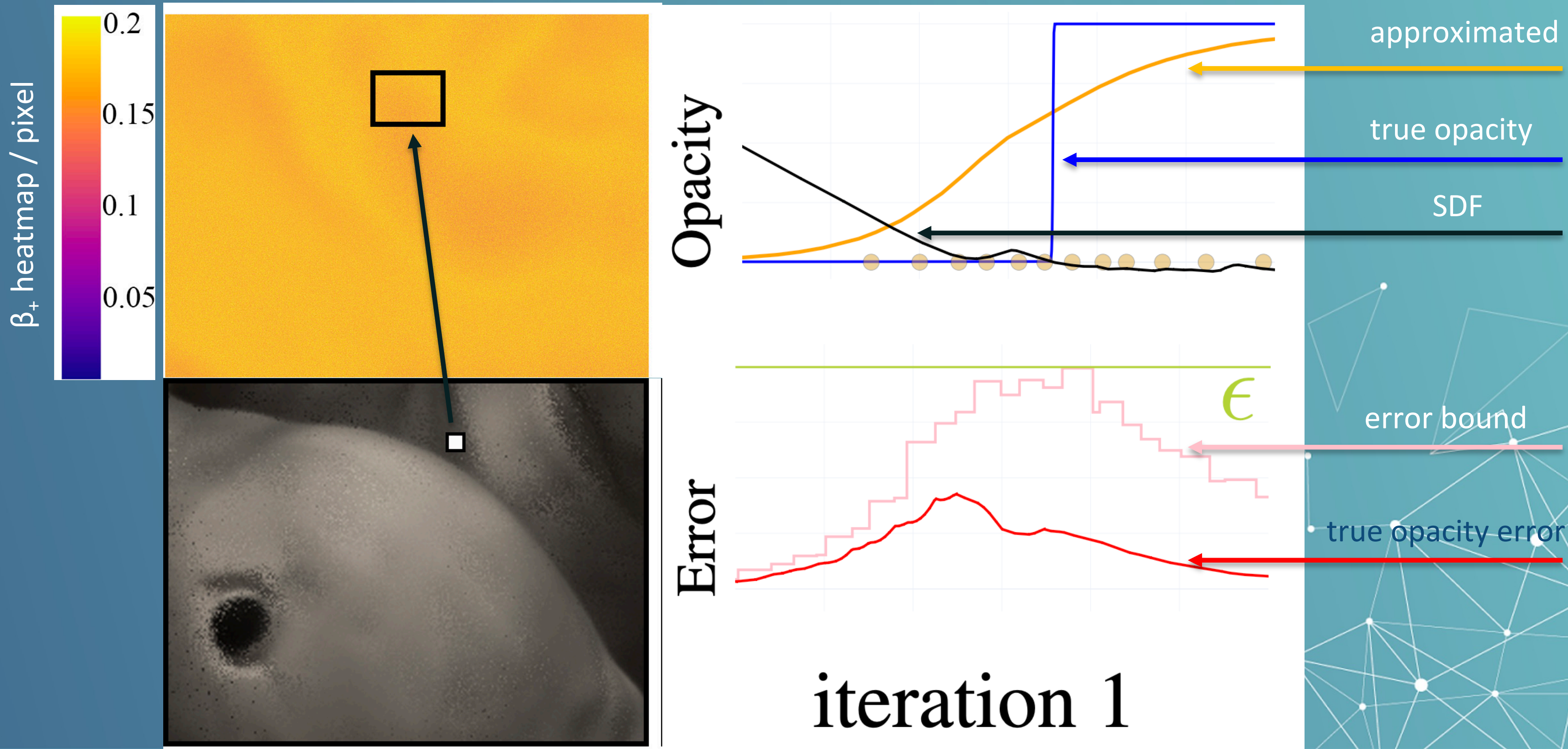
end

Estimate $\hat{O}$ using $\mathcal{T}$ and β_+

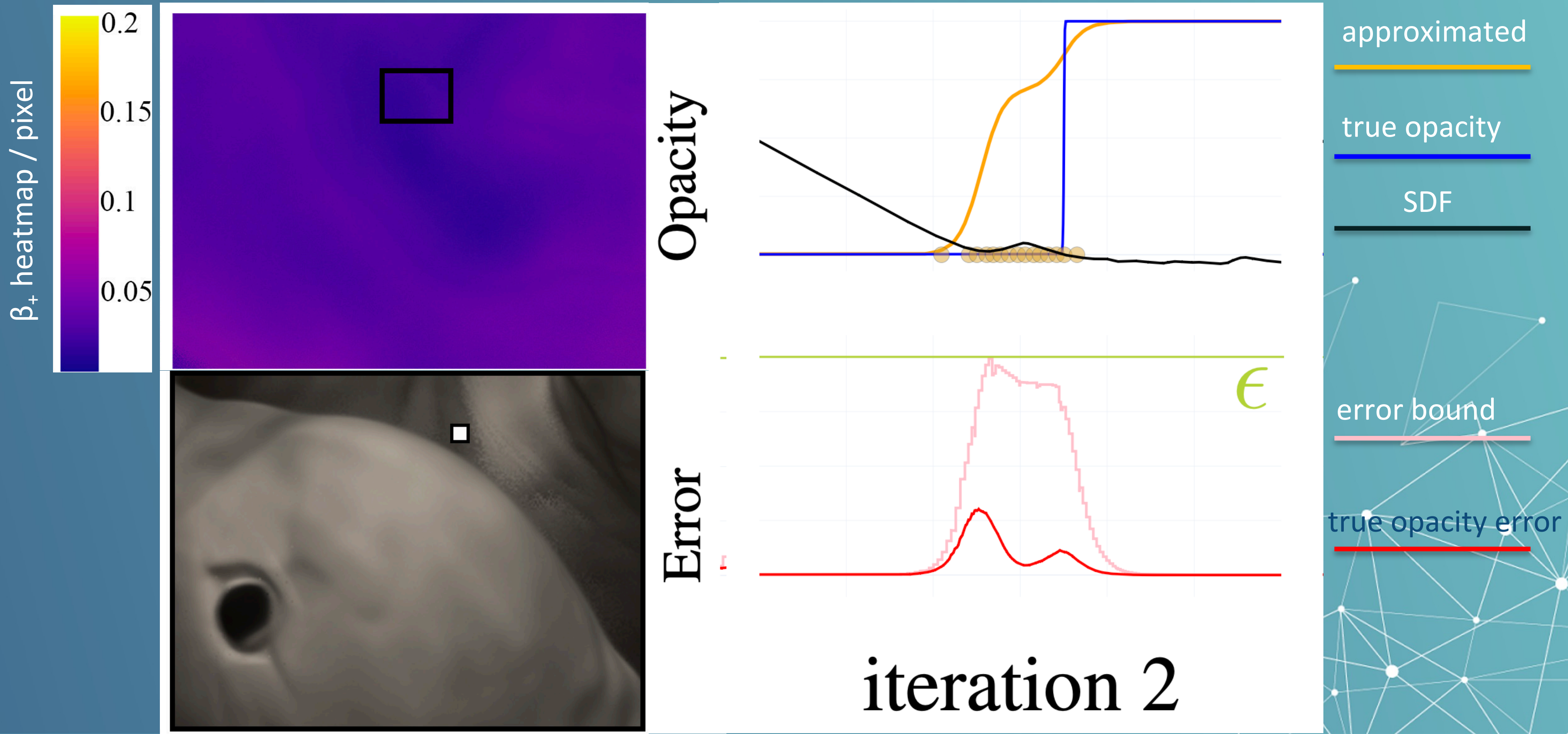
$\mathcal{S} \leftarrow$ get fresh m samples using $\hat{O}^{-1}$

return $\mathcal{S}$

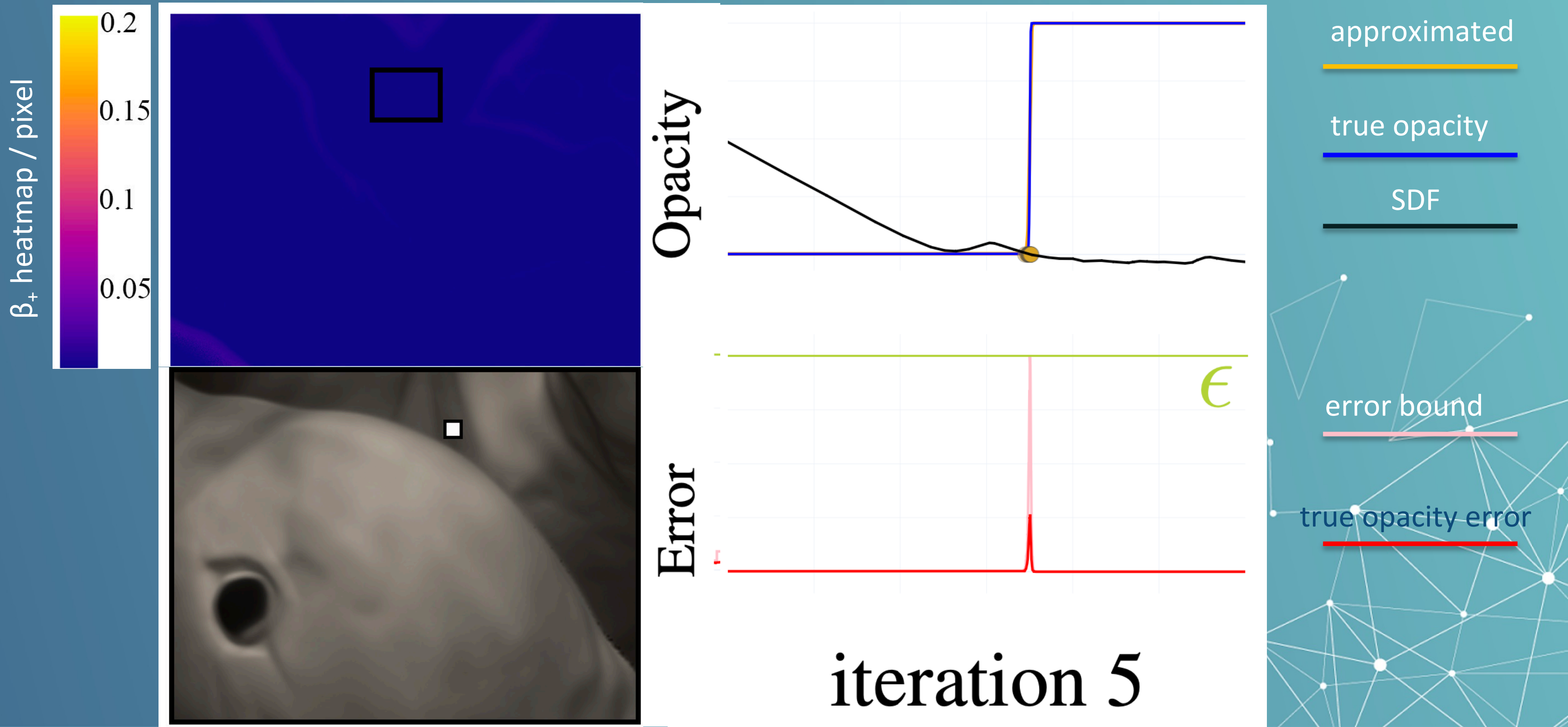
(4/5) Sampling Alogorithm - Qualitative



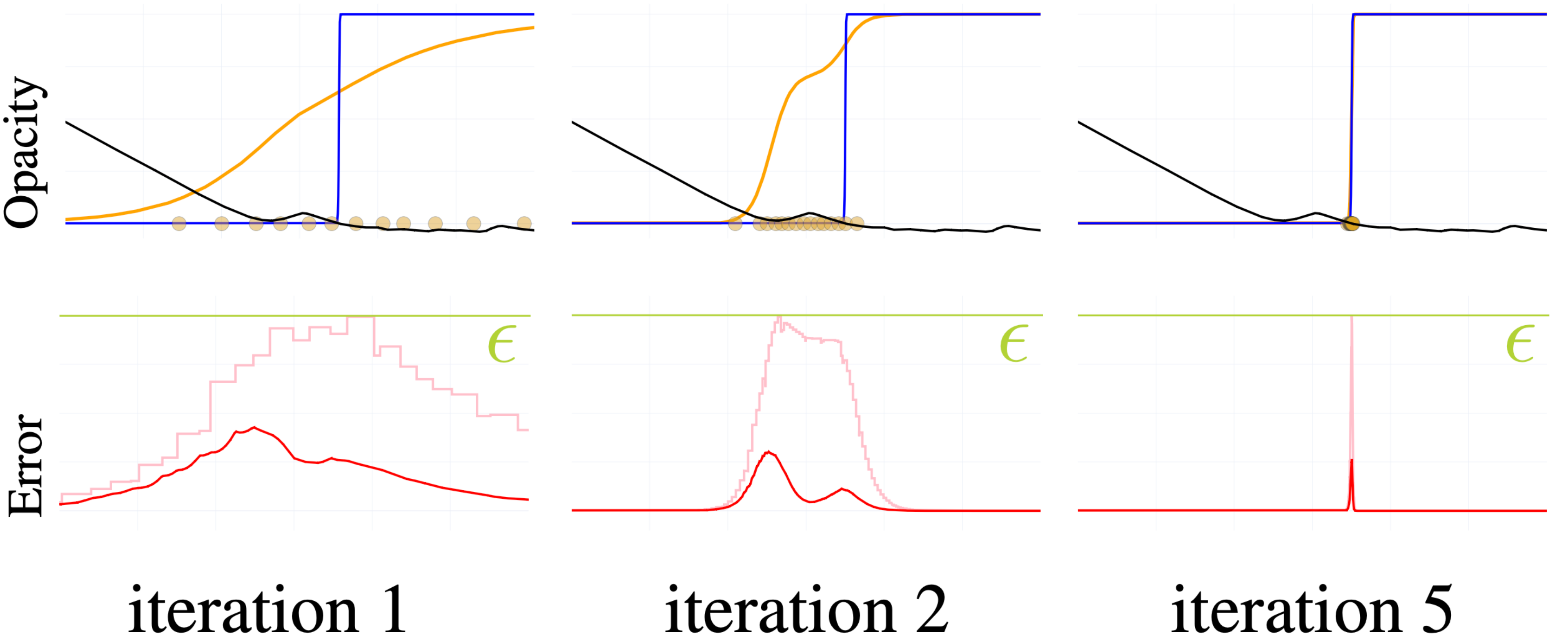
(4/5) Sampling Alogorithm - Qualitative



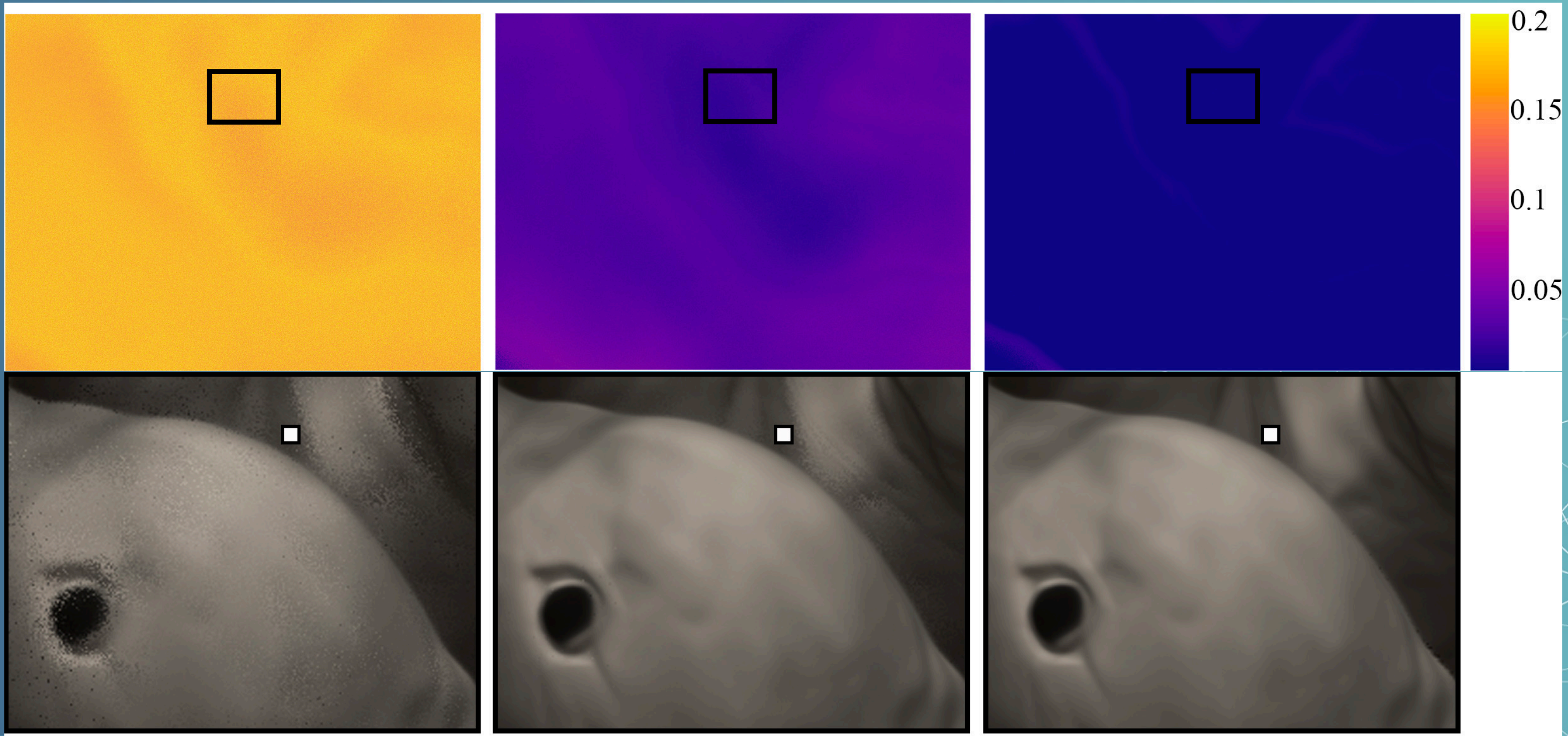
(4/5) Sampling Alogorithm - Qualitative



(4/5) Sampling Alogorithm - Qualitative



(4/5) Sampling Alogorithm - Qualitative



(5/5) Training

- 2x MLP:
  - Approximating the SDF of the learned geometry, and global geometry feature z of dimension 256 :
    - $f_\varphi(x) = (d(x), z(x)) \in \mathbb{R}^{1+256}$
  - Presenting the scene's radiance field with learnable parameters ψ :
    - $L_\psi(x,n,v,z) \in \mathbb{R}^3$
  - Two scalar learnable parameters
    - $\alpha, \beta \in \mathbb{R}$, with $\alpha = \beta^{-1}$
  - Positional enconding for x and v, same as NeRF

(5/5) Training

- For each pixel p a triplet $(I_p\ ,\ c_p\ ,\ v_p)$
  - $I_p \in \mathbb{R}^3$ is its intensity (RGB color)
  - $c_p \in \mathbb{R}^3$ is its camera location
  - $v_p \in \mathbb{R}^3$ is the viewing direction (camera to pixel)
- Training loss:
  - $\mathcal{L}(\theta) = \mathcal{L}_{\mathrm{RGB}}(\theta) + \lambda\, \mathcal{L}_{\mathrm{SDF}}(\varphi)$ (17)
  - $\mathcal{L}_{\mathrm{RGB}}(\theta) = \mathbb{E}_p \left\| I_p - \hat{I}_S(c_p, v_p) \right\|_1$ (18) color loss
  - $\mathcal{L}_{\mathrm{SDF}}(\varphi) = \mathbb{E}_z \left(\|\nabla d(z)\| - 1 \right)^2$ (18) Eikonal loss

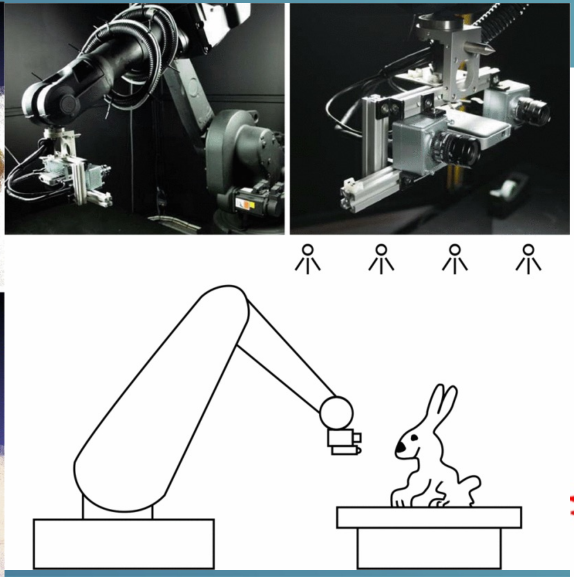
04

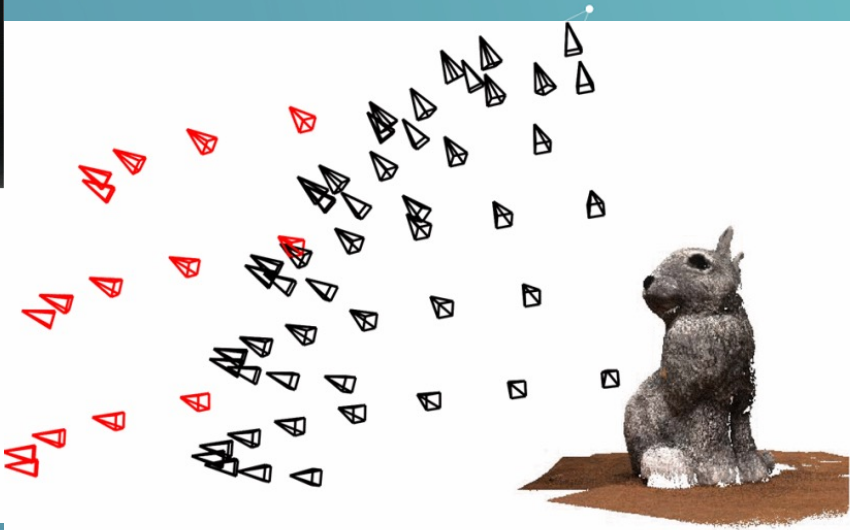
EXPERMENTS

Method evaluation on the challenging task of multiview 3D surface reconstruction

Multi-view 3D reconstruction

- Quantitative results for the **DTU dataset**
- DTU multi-view image; different objects; fixed camera and lighting parameters





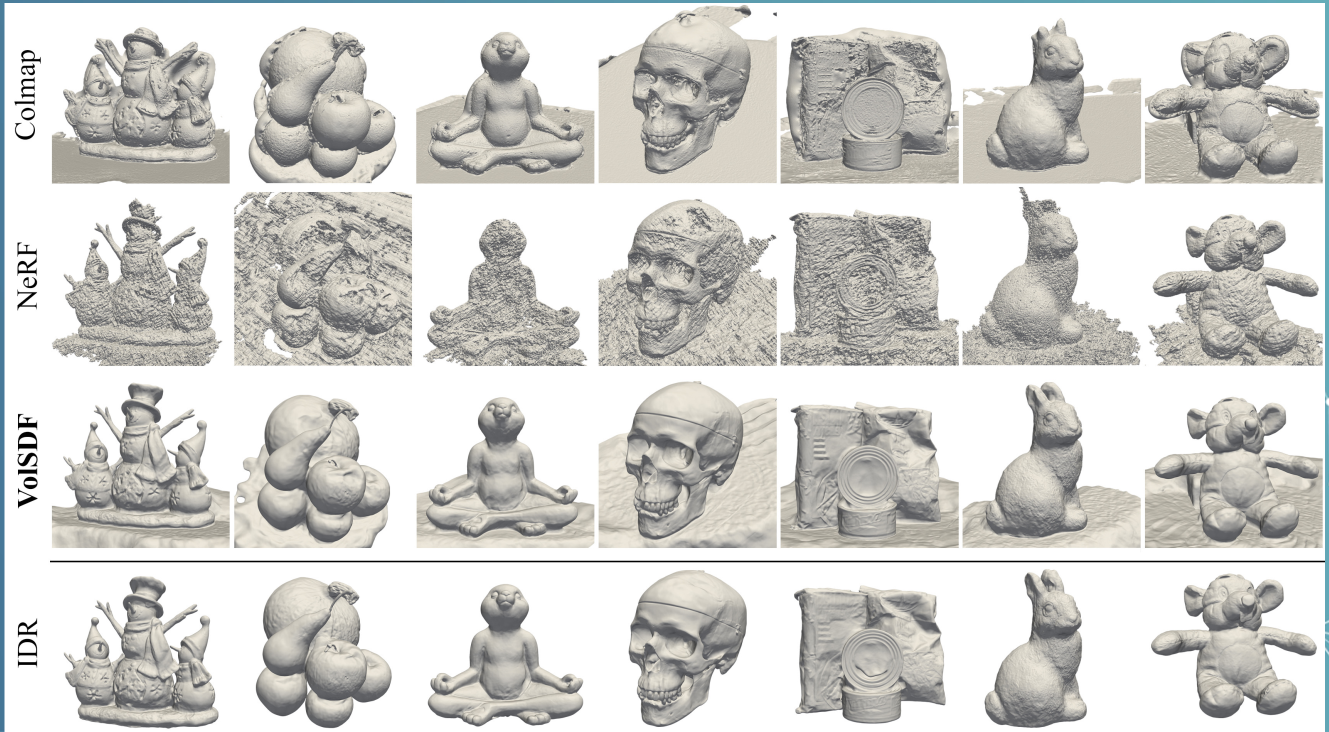
Multi-view 3D reconstruction

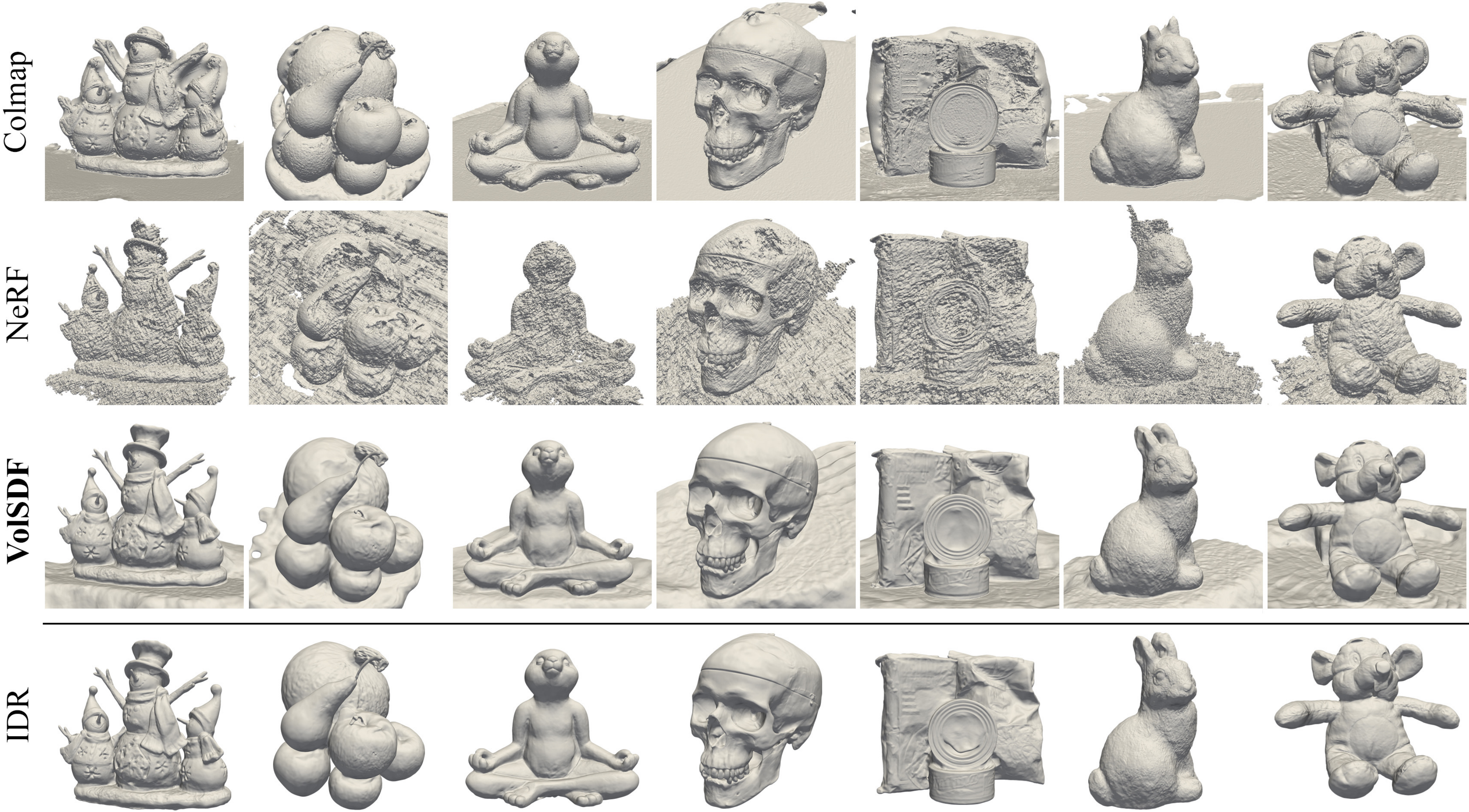
- Quantitative results for the **DTU dataset**
- DTU multi-view image; different objects; fixed camera and lighting parameters

	Scan	24	37	40	55	63	65	69	83	97	105	106	110	114	118	122	**Mean**
Chamfer Distance	**IDR**	1.63	1.87	0.63	0.48	1.04	0.79	0.77	1.33	1.16	0.76	0.67	0.90	0.42	0.51	0.53	0.90
	colmap$_7$	0.45	0.91	0.37	0.37	0.90	1.00	0.54	1.22	1.08	0.64	0.48	0.59	0.32	0.45	0.43	0.65
	colmap$_0$	**0.81**	2.05	**0.73**	1.22	1.79	1.58	1.02	3.05	1.40	2.05	1.00	1.32	0.49	0.78	1.17	1.36
	NeRF	1.92	1.73	1.92	0.80	3.41	1.39	1.51	5.44	2.04	1.10	1.01	2.88	0.91	1.00	0.79	1.89
	VolSDF	1.14	**1.26**	0.81	**0.49**	**1.25**	**0.70**	**0.72**	**1.29**	**1.18**	**0.70**	**0.66**	**1.08**	**0.42**	**0.61**	**0.55**	**0.86**
PSNR	**NeRF**	26.24	25.74	26.79	27.57	31.96	31.50	29.58	32.78	28.35	32.08	33.49	31.54	31.0	35.59	35.51	30.65
	VolSDF	26.28	25.61	26.55	26.76	31.57	31.5	29.38	33.23	28.03	32.13	33.16	31.49	30.33	34.9	34.75	30.38

Multi-view 3D reconstruction

- Qualitative results for the **DTU dataset**




Colmap
NeRF
VolSDF
IDR

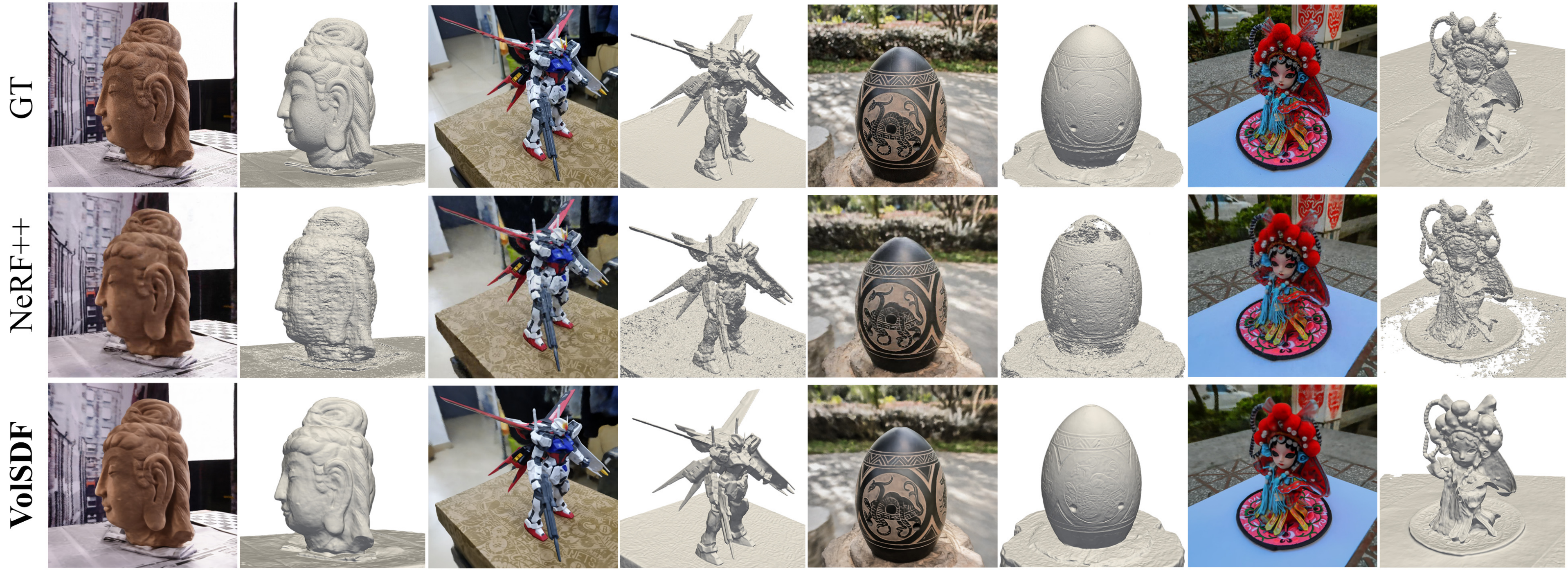
Multi-view 3D reconstruction

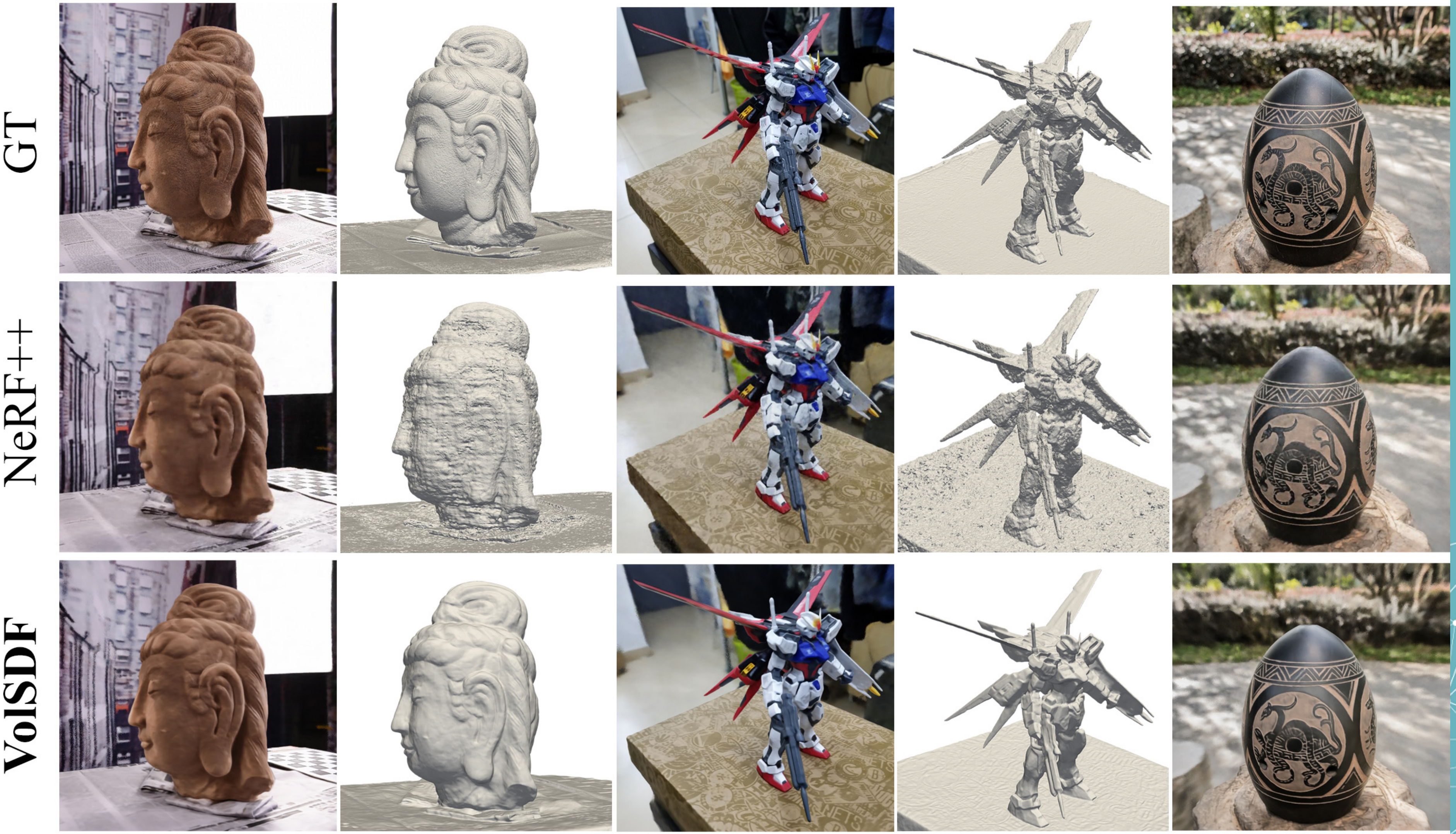
- Quantitative results for the **BlendedMVS** dataset
- Large collection of 113 scenes. High quality GT.
- 9 different scenes were selected

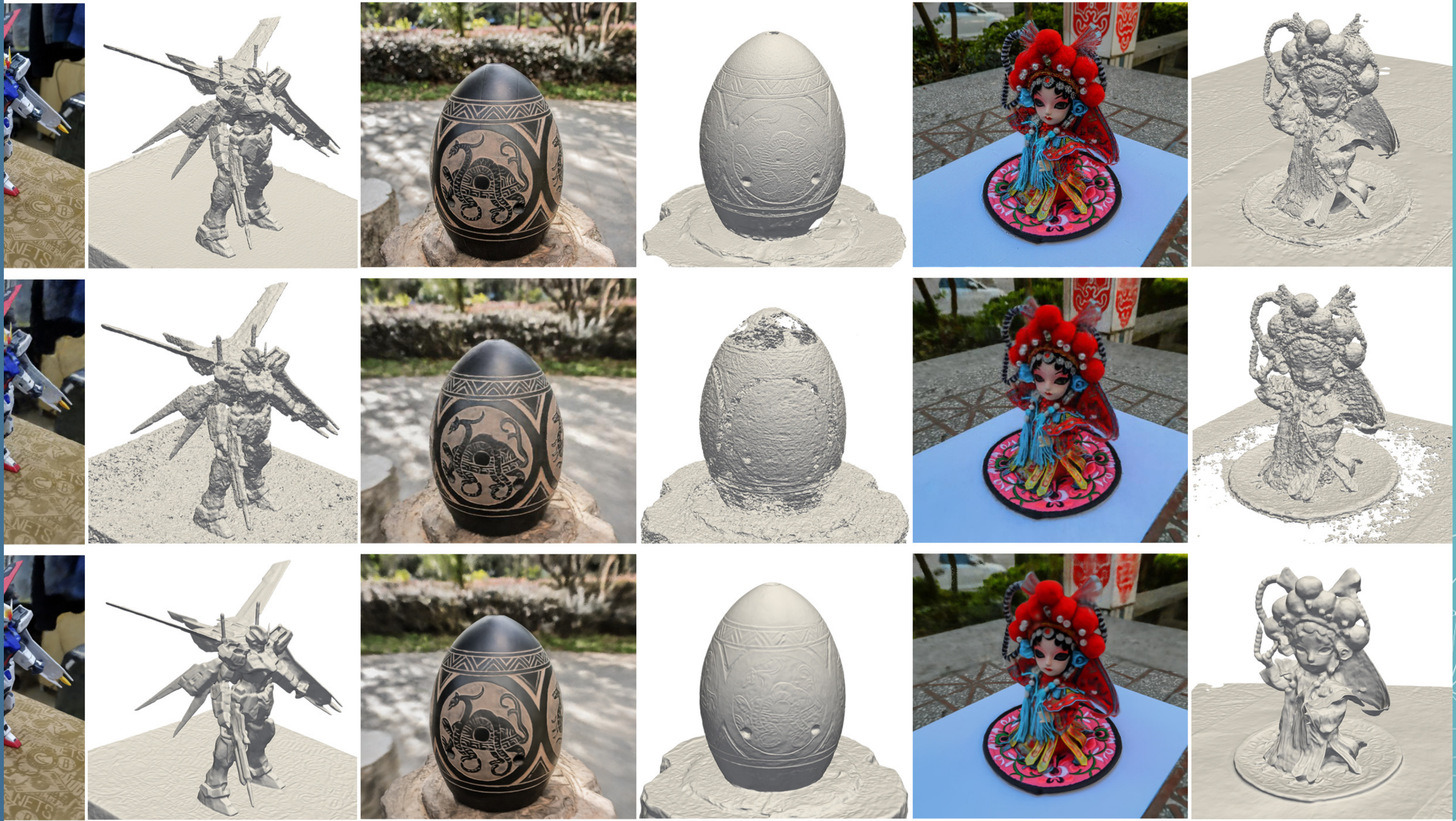
	Scene	Doll	Egg	Head	Angel	Bull	Robot	Dog	Bread	Camera	**Mean**
Chamfer l_1	Our Improvement (%)	54.0	91.2	24.3	75.1	60.7	27.2	47.7	34.6	51.8	51.8
PSNR	**NeRF++**	26.95	27.34	27.23	30.06	26.65	26.73	27.90	31.68	23.44	27.55
	VolSDF	25.49	27.18	26.36	29.79	26.01	26.03	28.65	31.24	22.97	27.08

Multi-view 3D reconstruction

- Qualitative results for the **BlendedMVS** dataset

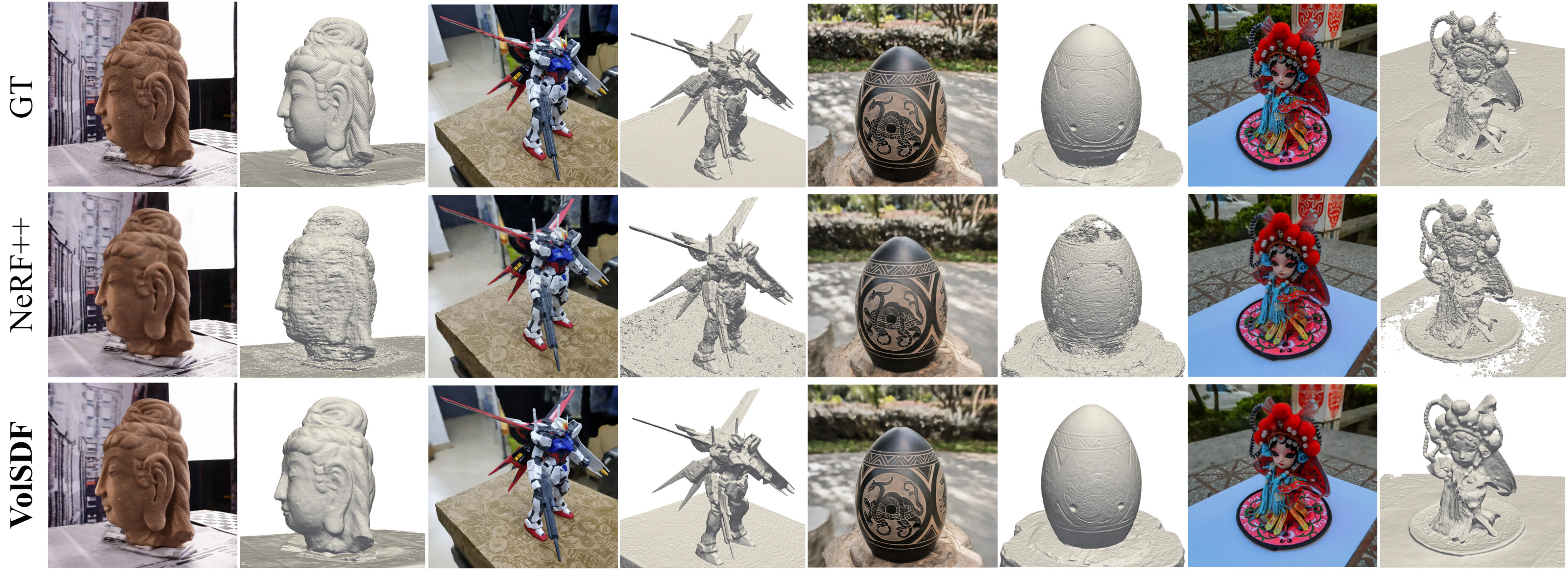



GT
NeRF++
VolSDF



Multi-view 3D reconstruction

- Qualitative results for the **BlendedMVS** dataset



05

CONCLUSION

You can enter here a subtitle if you need it

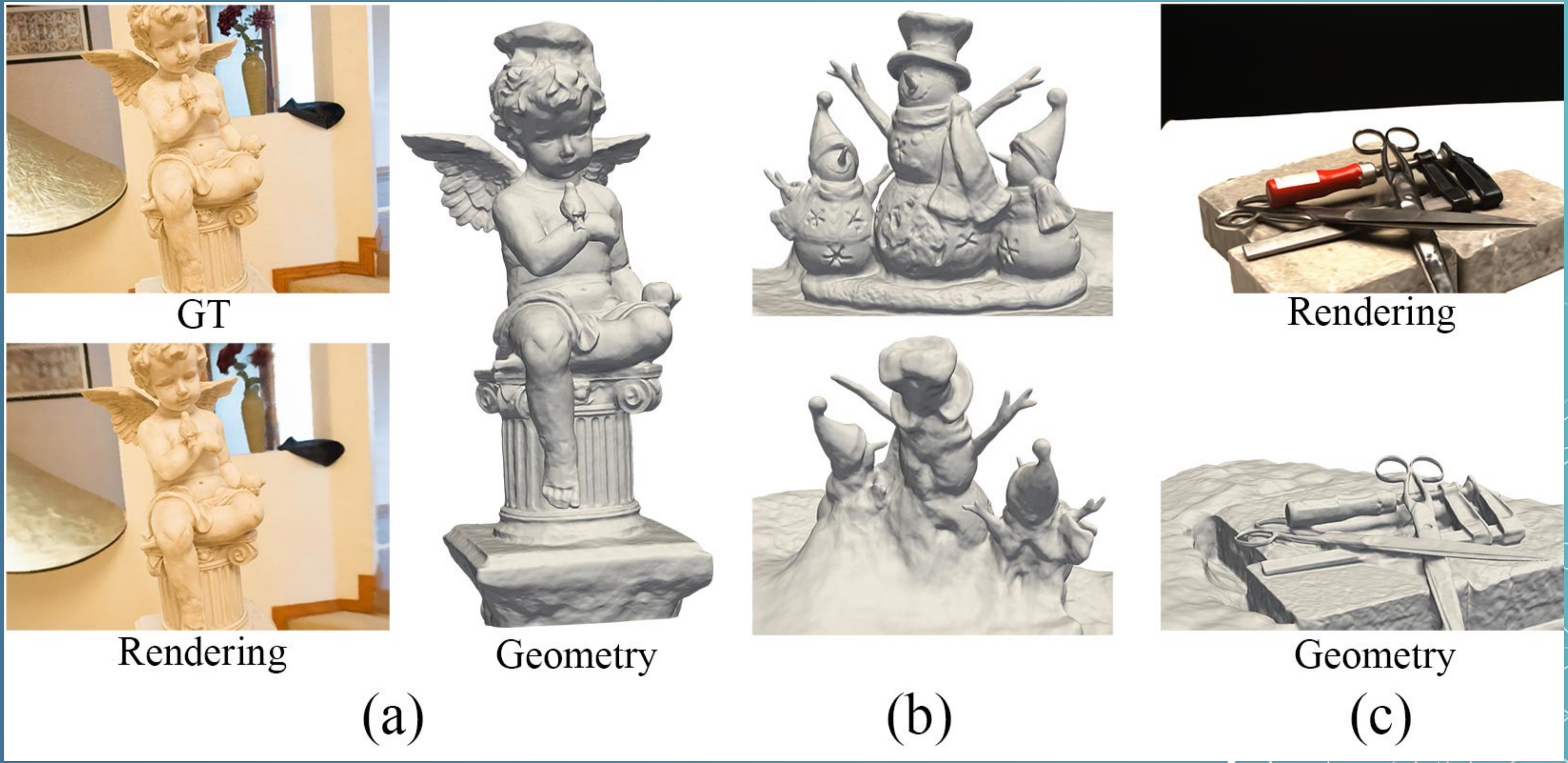
CONCLUSIONS AND REMARKS

- The paper does not have a proof of correctness for the sampling algorithm.
- Representing non-watertight manifolds and/or manifolds with boundaries, such as zero thickness surfaces, is not possible with an SDF
- Assumption of homogeneous density; extending it to more general density models would allow representing a broader class of geometries

CONCLUSIONS AND REMARKS

- High quality geometries can be learned in an unsupervised manner.
- Accurate geometry reconstruction from images can be used for malice purposes.

LIMITATIONS



06
SUPPLEMENTALS

You can enter here a subtitle if you need it



Thank you.