Fast Differentiable Transient Rendering for Non-Line-of-Sight Reconstruction

Markus Plack Clara Callenberg Monika Schneider Matthias B. Hullin
University of Bonn
Bonn, Germany
{mplack,callenbe,hullin}@cs.uni-bonn.de, moschn@uni-bonn.de

Abstract

Research into non-line-of-sight imaging problems has gained momentum in recent years motivated by intriguing prospective applications in e.g. *medicine and autonomous driving. While transient image formation is well understood and there exist various reconstruction approaches for non-line-of-sight scenes that combine efficient forward renderers with optimization schemes, those approaches suffer from runtimes in the order of hours even for moderately sized scenes. Furthermore, the ill-posedness of the inverse problem often leads to instabilities in the optimization.*

Inspired by the latest advances in direct-line-of-sight inverse rendering that have led to stunning results for reconstructing scene geometry and appearance, we present a fast differentiable transient renderer that accelerates the inverse rendering runtime to minutes on consumer hardware, making it possible to apply inverse transient imaging on a wider range of tasks and in more time-critical scenarios. We demonstrate its effectiveness on a series of applications using various datasets and show that it can be used for self-supervised learning.

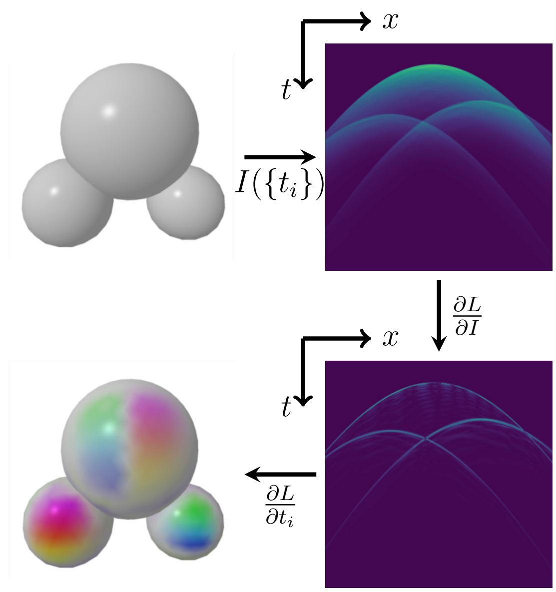


Figure 1. The triangle mesh $\{t_i\}$ is rendered into a transient image using a physically plausible forward model. After computing the loss, the gradient with respect to the pixel values is backpropagated onto triangle coordinates and their optional attributes. We show a false color visualization, where hue represents the direction and saturation the length of the xy gradients.

1. Introduction

Extending the vision beyond what is in the direct line of sight of an observer is a challenging problem with possible applications ranging from autonomous driving and robotic vision to safety and medical scenarios. Researchers have approached this non-line-of-sight (NLoS) imaging problem by pointing an ultrafast laser source at a wall which is in view of the observer as well as the hidden hidden target scene [35]. Using sensors that are able to resolve the travel time of the laser's light to observe reflections on the same wall, recording *transient images*, objects "around a corner" can be identified and further analyzed.

Many recent methods that use transient images for NLoS reconstruction represent the hidden scene as a volumetric albedo distribution [35, 9, 27]. While they are relatively fast and often yield convincing results, most of those approaches do not take important physical effects such as visibility/occlusion and surface normals into account. On the other hand, it has been proposed to reconstruct the hidden shape as a mesh using an analysis-by-synthesis approach, i.e., by making repeated forward simulations of light transport. Such methods are typically slow and need hours for the reconstruction [33, 11].

This work is inspired by the recent trend to solve inverse problems using task-specific differentiable renderers. The proposed differentiable renderer is specifically targeted to NLoS reconstruction. It extends the forward rendering ap-

Table 1. Comparison of relevant NLoS reconstruction approaches in terms of scene representation (**V**olume/**S**urface), usage of a physically-based image formation model (included ✔, somewhat included (✔), not part of the model ✘), their reconstruction time scales ranging from the order of milliseconds (**ms**) to hours (**h**) and their capability to generalize and adapt to new measurement geometries and higher resolutions, ranging from high (+) to intermediate (○) and to low/very low (−/−−) flexibility.

	Backprojection [35, 2]	(Directional) LCT [27, 41]	Occluders and Normals [8]	*f*-*k* migration [20]	Transient Rendering [11]	Surf. Optimization [33]	Deep NLoS [7]	Ours
Scene representation	V	V/S	V/S	V	S	S	S	S
Albedo reconstruction	✔	✔	✔	✔	✘	✘	✘	✔
Forward/Inverse Consistency	✘	✘	(✔)	✘	✔	✔	✘	✔
Normals, Occlusion	✘	(✔)	✔	✘	✔	✔	(✔)	✔
Reconstruction time	**s**	**s**	**h**	**s**	**h**	**h**	**ms**	**min**
Generalizability/adaptability	+	−	+	−	+	+	−−	+
Resolution	+	+	−	+	○	−	−	+

proach by Iseringhausen and Hullin [11] with additional degrees of freedom, such as surface albedo, and pairs it with an efficient implementation of the backward pass to backpropagate gradients to the parameters of the scene representation (Fig. 1). This enables the implementation of inverse solvers for a variety of NLoS sensing setups. A key feature of reconstructions obtained this way is that they are inherently consistent with a physically justifiable image formation model, a feature still missing in most recent reconstruction techniques.

We consider the following to be the main contributions of this work:

- We introduce a fast differentiable transient renderer for NLoS light transport. It extends an existing image formation model [11] by spatially varying albedo that is optimized jointly with the scene geometry in a simplified global optimization scheme.
- We demonstrate the effectiveness of the renderer for reconstructing NLoS scenes represented as radial basis functions and depth maps on simulated and real data. We further show that the framework generalizes to very high input resolution and object tracking tasks, thanks to its adaptability to irregular samplings and the use of stochastic optimization algorithms.
- We provide a complete PyTorch implementation of our renderer, along with the implementation of other NLoS reconstruction algorithms and various useful tools.[1]

Our framework runs on a consumer-grade GPU, and has proven to accept a wide range of input configurations. It can therefore serve as a portable and flexible development and test environment for future NLoS reconstruction approaches. We demonstrate this on the example application of a self-supervised network training that is based on our differentiable renderer.

2. Related Work

Transient/NLoS Imaging. Transient imaging allows to capture a scene's light response in space and time. Proposed originally by Abramson as early as 1978 using holographic techniques [1], it has become an increasingly relevant imaging modality with the development and growing accessibility of ultrafast photodetecting devices like streak cameras, single-photon avalanche diodes (SPADs) and photonic mixer devices (PMDs). A comprehensive overview of transient imaging advances can be found in [14].

In NLoS imaging, the light response of a scene is observed not directly, but via its reflection on a relay wall, while the target scene itself is outside the camera's view. Key tasks in this sensing mode are the reconstruction of position, shape and albedo of objects that are hidden both from direct illumination and observation. The reconstruction of NLoS scenes using transient data has been studied intensively using different types of measurement hardware, and different approaches exist in the literature [35, 39, 21, 3, 15, 9, 19, 26, 37, 36]. We compare the most important representatives by their different aspects and features in Table 1. Backprojection-based methods [35, 2] represent the hidden scene as a voxel grid and calculate a heat map of possible locations contributing to the measured space-time data, followed by a filtering step. Furthermore, Shen et al. [30] have proposed to optimize a neural transient field to reconstruct the hidden volume with arbitrary resolution. A different approximative approach, the light-cone transform (LCT) [27],

[1] `https://github.com/unlikelymaths/totrilib`

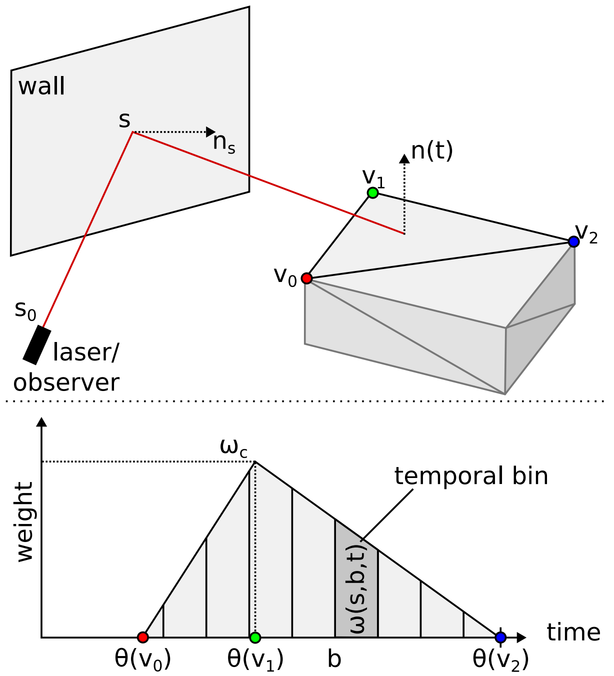


Figure 2. Coaxial measurement setup with the occluded scene represented as triangle mesh (top) and antialiasing of the corresponding temporal response using a trapezoidal filter (bottom).

provides a closed-form solution to the problem in a coaxial setup, where the relay wall is scanned in a regular grid with a beam-combined light source and detector. To reduce the acquisition time of transient images, circular sensing patterns have been proposed [12].

Since scenes represented as scattering density volumes by default do not support surface normals and occlusion effects, extensions with directional kernels [41] and iteratively adjusted linear weights [8] have been proposed. By modelling the light transport as the propagation of a (virtual) wave field, algorithms from wave optics and seismic tomography, like f-k migration, have successfully been adopted to solve the problem for regularly gridded input data [22, 20].

Instead of treating the hidden scene as a voxel-based albedo volume, several recent NLoS algorithms have introduced surface representations, for which physically justifiable light transport models are easier to achieve. After early attempts using planar walls [28], more recent approaches attempt to optimize triangle meshes and their reflectance properties by wrapping stochastic [33] or deterministic [11] renderers a task-specific optimization scheme. The renderer proposed in this paper builds upon the model by Iseringhausen and Hullin [11] and achieves significantly improved reconstruction times by introducing analytical derivatives and utilizing a modern deep learning infrastructure.

Lastly, the availability of large amounts of synthetically generated data has enabled the training of feed-forward networks for the NLoS reconstruction problem for surface-oriented [7], volumetric [4, 25] and implicit [6] scene representations.

Differentiable Rendering. In the case of direct-line-of-sight inverse rendering a number of studies have investigated approaches to compute the gradient of the visibility between two points, which is not differentiable as it is either 0 or 1. This is especially problematic as those gradients are needed to properly move edges across pixels/the visible hemisphere of a surface. One of the first general approaches was published by Li et al. [18]. They compute the gradient through Monte Carlo sampling rays along the edges of triangles. More recently, Zhang et al. [42] have proposed a method to directly differentiate path integrals through a reparametrization. However, in line with the work of Tsai et al. [33], we do not take visibility gradients into account, as the computation would increase the complexity. We still demonstrate that our method works even for cases where occlusion happens in the scene.

In the setting of transient imaging, various approaches have been proposed to address the forward rendering problem [32, 13, 31, 24] and to model sensors for accurate simulation of transient images [10]. General differentiable renderers such as [40, 38] aim to facilitate analysis-by-synthesis reconstruction approaches. However, their universality comes at the cost of computational complexity and they suffer from long runtimes even in cloud computing environments. By restricting the image formation model to the three-bounce NLoS setting, our renderer runs fast on consumer-grade GPUs with moderate amounts of memory.

3. Differentiable Transient Rendering

The key part of our method is the formulation of the transient image formation model as a differentiable function and the efficient backpropagation of gradients through the renderer. We discuss the forward model and the gradient computation in Section 3.1. To increase stability of optimization problems on measurement data, we propose to add a background network in Section 3.2.

3.1. Image Formation

Our image formation model follows that by Iseringhausen and Hullin [11]. Here, we recall it for the coaxial capture geometry, where laser and detector are combined in a single beam, before outlining the computation of gradients. More detailed gradient equations, special cases, and their derivation for both coaxial and independent scanning geometries are given in a supplemental document.

Forward Model. Fig. 2 depicts the measurement setup that is approximated by our renderer and a visualization of the distribution of the recorded light into temporal bins of the time-resolved detector. As interreflections on the object contribute little to the rendered transients, we follow the common three-bounce assumption which only takes light paths into account that move from the laser source s_o to a point on the wall s, onto a triangle $t = (v_0, v_1, v_2)$ of the object surface, back to the wall point s, and are recorded by the time-resolved sensor that is collocated with the laser at s_o.

We approximate the incoming radiance for each triangle by the constant radiance of the triangle centroid $c(t)$ over the full area of the triangle as

$$\alpha(s,t) = f(s \to c(t) \to s)\eta(s \to c(t))\eta(c(t) \to s)A(t), \tag{1}$$

where f denotes the BRDF, $\eta(x \to y)$ the geometric coupling between the two points x and y, and A the area of the triangle. Using $n(t) = (v_1 - v_0) \times (v_2 - v_0)$ as the unnormalized normal vector of the triangle, and n_s as the surface normal of the wall at s, and further assuming Lambertian reflection with albedo $a(t)$, the full expression for α can be simplified to

$$\alpha(s,t) = a(t)\frac{\langle n_s, c(t) - s\rangle^2 \langle n(t), c(t) - s\rangle^2}{\|n(t)\|\|c(t) - s\|}. \tag{2}$$

However, Lambertian reflection is no restriction of our method and any differentiable BRDF model can be used. We have removed the visibility term from α for ease of notation as it is not differentiable, but still perform a visibility check $\nu(s, c(t))$ between the triangle centroid and the wall as seen in Eq. (6).

To compute the total irradiance contributed by a triangle to each transient bin b, $\alpha(s,t)$ is distributed according to a weighting function $w(s,t,b)$ as shown in Fig. 2 according to the length of the light paths and hence the time of flight. Assuming rectified measurements, the corresponding bin of each vertex is given by

$$\theta(v_i) = (2\|v_i - s\|_2 - \phi)/\delta, \tag{3}$$

where ϕ denotes the offset and δ the bin width of the scanning setup. Note that θ is not an integer value and as such is differentiable. We assume that the vertices are sorted in ascending order of total distance. The weight at the center is given as

$$\omega_c(t) = \frac{2}{\theta(v_2) - \theta(v_0)}. \tag{4}$$

For the bins that fall between the points $\theta(v_0)$ and $\theta(v_1)$, we compute the weight as the area under the left triangle as

$$\omega(s,b,t) = \left(b + \frac{1}{2} - \theta(v_0)\right)\frac{\omega_c(t)}{\theta(v_1) - \theta(v_0)}. \tag{5}$$

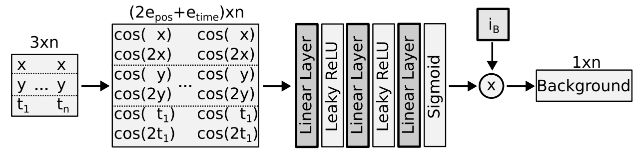


Figure 3. Architecture of our background network. The position of the scan points and the temporal bin are encoded using cosine terms (two each in this example), followed by a linear neural network operating on the first dimension and a scaling. Layers with learnable parameters are highlighted.

The equation for weights between $\theta(v_1)$ and $\theta(v_2)$ follows analogously. The full rendering function of a set of n triangles can be written as

$$I(\{t_0, \ldots, t_{n-1}\}) = \left(\sum_{i=0}^{n-1} \nu(s, c(t_i))\alpha(s, t_i)\omega(s, b, t_i)\right)_{s,b} \tag{6}$$

Backpropagation. To avoid the need for numerical derivatives [11], we explicitly compute gradients through backpropagation of the gradient of a loss function $L(I)$. During the backward pass, we evaluate

$$\nabla_{t_i} L = \sum_s \sum_b \frac{\partial L}{\partial I_{s,b}} \nabla_{t_i} I_{s,b} \tag{7}$$

for each triangle t_i. We can reformulate this as

$$\begin{aligned}\nabla_{t_i} L = \sum_s v(s, t_i)\Bigg(\nabla_{t_i}\alpha(s,t)\sum_b \frac{\partial L}{\partial I_{s,b}}\omega(s,b,t) + \\ \alpha(s,t)\sum_b \frac{\partial L}{\partial I_{s,b}}\nabla_{t_i}\omega(s,b,t)\Bigg).\end{aligned} \tag{8}$$

The gradient of α can be computed using logarithmic derivatives as shown in the supplemental document. In order to efficiently evaluate the gradients, we implement all computations as NVIDIA Optix programs. This enables us to directly continue with the radiance/gradient computation after the visibility test. Note that there is no need to evaluate the full sums in Eq. (8), but only the subset between the bins $\theta(v_0)$ and $\theta(v_2)$ which are evaluated first.

3.2. Background Model and Reconstruction Loss

Even though the formulated model is physically motivated, inconsistencies with real measurements can be expected. This can be due to approximations or in the case where the true BRDF is different from the model. More prominently, there can be background illumination, for instance from other surfaces that are not part of the scene. Those effects would lead to incorrect gradients and reduce the quality of the reconstruction.

To remedy the influence of such effects, we propose to add a background prediction network (Fig. 3) to the optimizations that use the differentiable rendering proposed above. The network takes each scan position (x, y) together with the temporal position t_i and transforms them into positional and temporal encodings using cosines similar to the approach originally proposed by Vaswani et al. [34]. Those encodings are passed through a simple neural network to produce a transient response. To improve performance, the temporal resolution is reduced by a factor of 8 and the transient image produced by the network is linearly upsampled to the final resolution.

We also add a condition to prevent the transient background from capturing too much of the true image as follows. The output of the network $I_B \in (0,1)^{S\times B}$ is scaled using an intensity value i_B that is part of the network parameters. Defining the average power of the transient spectra $P(I)$ we add the condition

$$P(I_B) \leq \lambda_I P(I_R), \quad P(I) = \frac{1}{S}\sum_{i=0}^{S-1} \|I_{i,:}\|_2 \tag{9}$$

which we enforce by clamping i_B appropriately after each optimization step, where I_R is the rendered transient image of the current iterate. The parameter λ_I can be used to control the total amount of light in the transient background. For most of our experiments we set it to 1, which we found to work well.

The benefit of using such a network is that it is independent of the arrangement of scan and laser points and that both sharp jumps as well as smooth gradients can be represented, depending on the input and the effects easily captured by our forward model.

We formulate the reconstruction loss as

$$L(\rho, \phi) = \min_{\gamma} \|\gamma(I_R(\rho) + I_B(\phi)) - I_{\text{in}}\|_2, \tag{10}$$

where ρ is the scene parameterization, ϕ the parameters of the background network, and γ the unknown scaling between the input and the reconstruction. For the optimization of depth maps in Section 4.2 we add γ to the set of parameters after initializing it appropriately. Unfortunately, we found that this approach is problematic in the case of radial basis function optimization as the addition and the removal of blobs can lead to a significant change in the transient image. Instead, we replace γ with the minimizer of Eq. (10).

An extension to other loss functions, that more accurately represent the noise model of transient images, is possible, but similar to [33] we found $L2$ loss to work well over a large range of datasets.

4. Applications

To demonstrate the effectiveness of our implementation, we show its application on three different parametrizations of the geometry used for reconstruction (Section 4.1 and Section 4.2) and tracking (Section 4.3) of hidden objects. In addition, we show that our method can also be used for self-supervised training in Section 4.4.

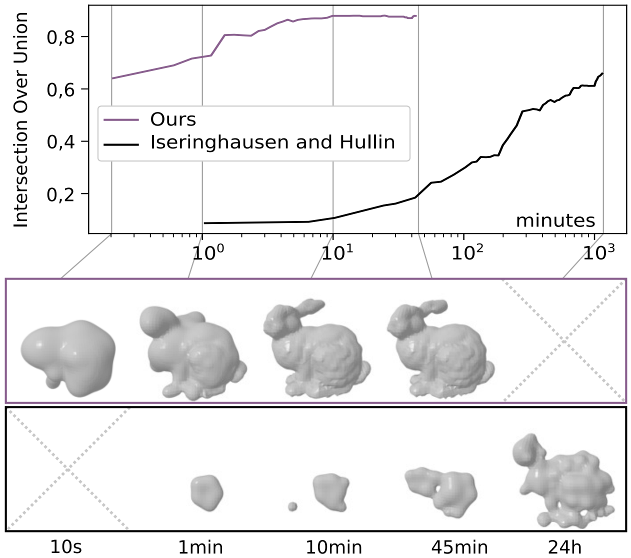


Figure 4. Runtime comparison with the baseline method of Iseringhausen and Hullin [11]. Both methods yield accurate meshes with a mean absolute depth error of 2.91cm (ours) and 2.98cm (baseline) at the end of the optimization for this synthetic 2x2m scene.

We evaluate our method on common datasets using both simulated data from [5] and our own renderer, as well as measurements from [35], [20], and [27].

4.1. Radial Basis Function Approximation

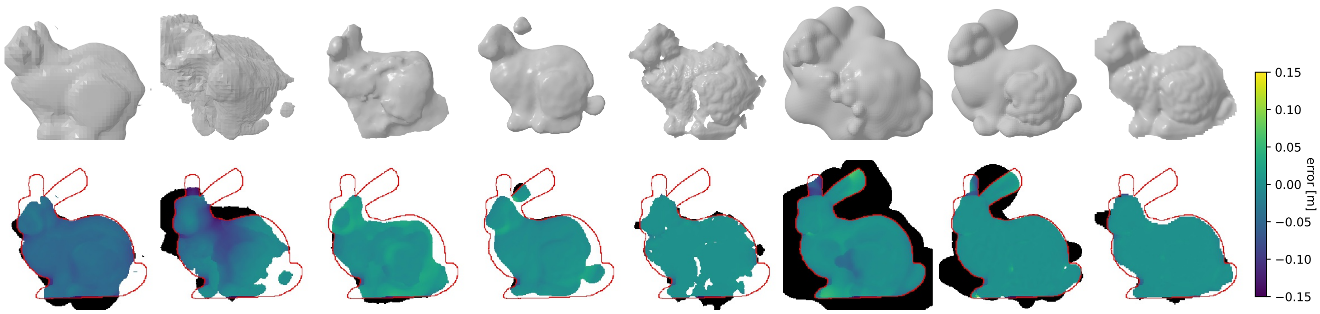
As a direct optimization of triangular meshes is difficult due to e.g. self intersections, we follow the approach of [11] and optimize a set of radial basis functions that approximate the density inside a volume. We generate a mesh by extracting the isosurface using a differentiable marching cubes [23] implementation.

For a set of Gaussian basis functions f_i with parameters p_i and σ_i the density at a position $x \in \mathbb{R}^3$ is given as

$$d(x) = \sum_i f_i(x), \quad f_i(x) = e^{-\frac{\|x-p_i\|_2}{2\sigma_i}}. \tag{11}$$

Additionally, we allow the basis functions to carry attributes such as an albedo value. This yields another volume by computing the weighted average of the attribute values. Those values are interpolated along with the vertex positions in our implementation of the marching cube algorithm.

Note that in this scenario, the computational complexity of the derivative of the rendering as well as the marching cubes step does not depend on the number of radial basis



(a) FBP [35] (b) $f - k$ Migration [20] (c) Fermat [39] (d) D-LCT [41] (e) Tsai et al. [33] (f) Rbf (Sinogram) (g) Rbf (h) Depth Map

Figure 5. Reconstructions of the simulated bunny from [5] of various methods (a–e) compared to our results (f–h). The first row shows the resulting meshes and the second row plots the corresponding depth errors between the respective reconstructions and the ground truth.

functions. Therefore, the iterative algorithm of [11] can be adapted to allow an optimization of all basis parameters in all steps, because there is less need to reduce the number of derivatives that are computed. Additionally, we add another sampling of new blobs that is focused on modifying the surface of the mesh. By backpropagating the current loss to the vertices, we add new blobs at the vertex positions with probability proportional to the length of the vertex gradients. We reduce runtime of the optimization by choosing a rough resolution at the initial iterations and doubling the resolution at certain intervals. More details are given in the supplemental document.

We demonstrate the runtime improvement of our method over the baseline of Iserinhausen and Hullin [11] in Fig. 4. Both methods reconstruct the same synthetically rendered mesh on the same hardware setup. Our method yields convincing results after a few minutes, while the baseline method takes a full day to produce a recognizable solution.

To further evaluate the correctness of our model we use the simulated bunny data from [5] and compare our results qualitatively (Fig. 5) and quantitatively (Table 2) against various other reconstruction methods. To convert volumetric reconstructions into a mesh we use marching cubes [23] and search for a threshold that maximizes the intersection over union (IoU). While our GPU implementations of those methods run much faster, we found that the quality of the results deteriorates quickly when using lower resolution input. At the same time, we needed to use a scanning resolution of 64×64 for a fair comparison with the method of Tsai et al. [33], which also uses differentiable rendering, but is much slower than our method.

While our Rbf-based reconstruction overestimates the shape of the bunny, it manages to reconstruct one ear and the overall shape very accurately, which is confirmed by a IoU value that is only surpassed by our depth map based reconstruction shown in the next section. We also include results for a reconstruction from a transient sinogram as proposed by [12], where the overall shape is even larger, but it still yields convincing results and an error comparable with volume based methods even though only 8.7% of the transient spectra are used.

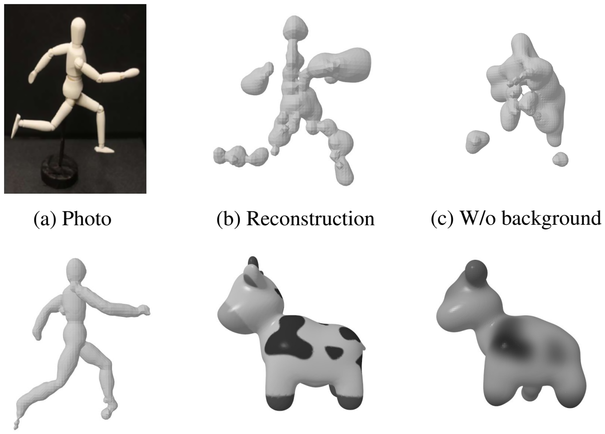
We test the reconstruction of objects with spatially varying albedo on the Spot model and show results in Fig. 6. Although the albedo information is associated with the radial basis functions and not provided as a high-resolution texture, simple changes in albedo are faithfully reconstructed, as can be seen with features like the cow model's dark spots and hooves.

We also demonstrate the application of out method on real data using the mannequin measurements of Velten et al. [35] and show the reconstructions in Fig. 6 along with a reconstruction using a rendered mannequin using the same setup. The overall shape of the reconstruction matches the mannequin from the reference, even though details are lacking when compared to the synthetic reconstruction. As the data was acquired using a non-confocal setup, there are only a few methods that can reconstruct such a measurement. Figure 6 also highlights the ability of our background network to deal with an arbitrary scanning setup and its importance for the reconstruction.

4.2. Depth Map Optimization

In this example application, we optimize the vertex positions similarly to [33]. To remove the need for additional mesh operations we restrict the optimization of the position to the depth values of a grid, i.e. only the z-coordinate is optimized. As such an object would lead to a large amount of unwanted background we also optimize the albedo of the vertices.

To improve stability of this approach we opt to add a total variation regularization [29] to our loss. We regularize both the color attribute as well as the depth. As the color values $c \in [0,1]^{H \times W}$ are naturally bounded to the $[0,1]$ interval, we choose to limit the depth map $d \in [0,1]^{H \times W}$



(a) Photo (b) Reconstruction (c) W/o background

(d) Reconstruction (e) Spot Model (f) Reconstruction

Figure 6. Reconstructions of measured [35] (a–c) and a synthetic mannequin dataset [11] (d), and a reconstruction of the "Spot" model (synthetic), represented using radial basis functions with spatially varying albedo (e,f).

Table 2. Quantitative comparison of reconstructions from the simulated measurements of the bunny [5] with various other methods showing the runtime (minutes:seconds), intersection over union (IoU, higher is better), as well as mean absolute error (MAE, lower is better) and root-mean-square error (RMSE, lower is better) in cm. For each metric, the best value is highlighted in red and the best follow-up in blue.

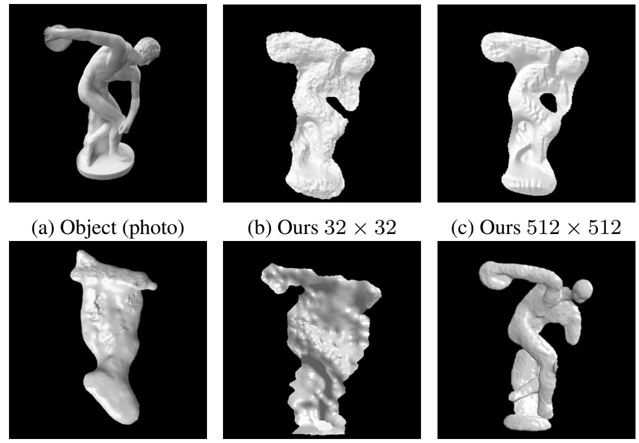
Method	Runtime	IoU	MAE	RMSE
FBP [35]	<0:01	0.738	4.86	5.03
f–k [20]	<0:01	0.659	3.81	4.86
Fermat [39]	0:12	0.730	1.05	1.58
D-LCT [41]	0:05	0.728	0.59	0.95
Tsai et al. [33]	102:06	0.730	0.28	1.03
Rbf	4:51	0.760	0.41	1.33
Rbf (Sinogram)	1:34	0.490	1.13	2.10
Depth Map	2:25	0.803	0.26	0.76

to the same interval and apply a scaling and translation to the reconstruction volume before the rendering. Hence, our loss function can be written as

$$L(c, d) = \|I - R(c, d)\|_2 + \lambda_d TV(d) + \lambda_c TV(c), \quad (12)$$

where TV is an isotropic total variation with $\epsilon = 0.001$ for smoothing with regularization weights λ_d and λ_c. We initialize with a coarse resolution depth map and double the resolution during the optimization.

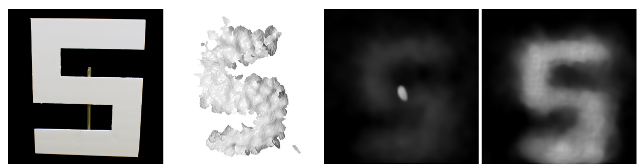
We also evaluate this representation on the synthetic bunny from [5] in Fig. 5. The reconstruction captures the fine details of the surface structure better than all other representations, resulting in the best metrics as listed in Table 2. While D-LCT [41] runs much faster, it lacks some details when compared to differentiable rendering based approaches. At the same time our method offers a significant runtime improvement over the method of Tsai et al. [33].



(a) Object (photo) (b) Ours 32×32 (c) Ours 512×512

(d) Fermat [39] (e) Tsai et al. [33] (f) D-LCT [41]

Figure 7. Reconstruction of the "Statue" dataset photo shown in (a) [20]. (d)–(f), three reconstructions from recent literature (adapted from [41]). (b) and (c) show reconstructions obtained from our framework using a depth map representation for different input resolutions.



(a) Object (b) Tsai et al. (c) Ours raw (d) Ours flat-field

Figure 8. Reconstructions of the "Diffuse S" dataset [27]. From left to right: photo of the object [27] (a); reconstruction by Tsai et al. [33] (b); reconstructions using our method as depth map with varying albedo: (c), raw dataset; (d), flat-field corrected dataset.

We show the application of this approach on measurement data of a statue [20] in Fig. 7 and the diffuse S [27] in Fig. 8. The quality of the reconstructions of the statue is on par with the reconstruction of D-LCT from [41]. Even after reducing the resolution down to 32×32 the quality stays consistent with a reconstruction time of only 39 seconds. For higher resolutions, we switch to a stochastic gradient descent optimization with batch size of 4096 scan points. Therefore, the reconstruction time does not increase beyond a resolution of 64×64 and keeps below three minutes.

The reconstruction of the diffuse S shows a failure case of our background network, which cannot deal with the large amounts of spatially varying background present in the dataset. We clean the data up by applying a semi-automatic flat field correction that estimates a static background component from the signal-less portion of the dataset (before the first transient onset). The resulting reconstruction is similar to the one of Tsai et al. [33], but runs in under three minutes.

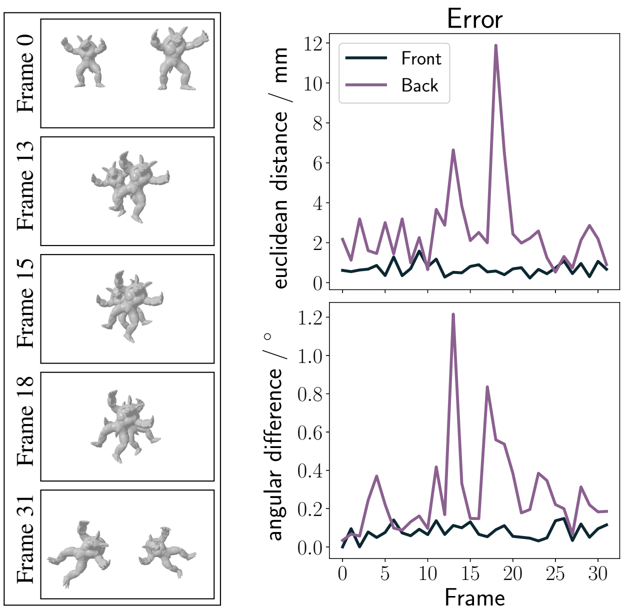


Figure 9. The two armadillos are positioned with 1 m and 1.5 m distance to the wall and perform a linear motion and rotation as indicated by the key frames in the first column. The transient input has a PSNR of 28.4. The plots on the right show the position and rotation error in millimeters and degrees, respectively.

4.3. Tracking

This application takes as input one or more meshes of hidden objects and a transient image of these objects at unknown positions. The aim is to infer the hidden object's spatial position and orientation. To this end, we optimize the position vector and the orientation quaternion of each object to match the given transients.

We demonstrate the tracking of two armadillo meshes over a video in Fig. 9. The first frame is initialized to the correct position and rotation and we iteratively optimize the transformation of both objects for each frame using the results of the previous frame as an initialization.

The positions and rotations are matched with negligible errors for both objects. The accuracy of the armadillo in the back is slightly lower because of the reduced light intensity reaching the wall, and it degrades during the middle of the video where most of the object is occluded by the armadillo in the foreground. The estimation quality is, however, still reasonable even though our method only approximates the full visibility of the triangles and does not compute gradients for the visibility term. The optimization of a single transform with more translation and rotation is shown in the supplemental document.

4.4. Proof of Concept: Self-Supervised Learning

Finally, we demonstrate the flexibility of our differentiable renderer by using it to train a reconstruction network in a purely self-supervised manner. We generate synthetic data from random sets of gaussian blobs similar to Section 4.1. The convolutional network takes the transient image as input and outputs a density volume that is converted into a mesh using marching cubes. We pass this mesh through our differentiable renderer and compute the L2 loss between the resulting transient image and the network input, which can be backpropagated through all steps to update the network parameters.

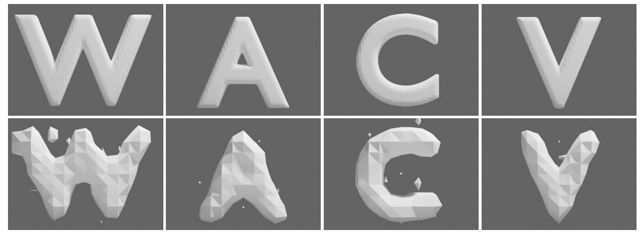


Figure 10. Ground truth models (top) and their reconstructions (bottom) using a network trained in a self-supervised regime with synthetic data generated from volumetric blobs.

We train the network for 500000 iterations using Adam [16] with a batch size of 32. The volume and scan point resolution is set to 16. Additionally, we add a small L2 regularization of the gradients of the volumetric output for smoothness. Results are shown in Fig. 10.

5. Conclusion

We have demonstrated that an efficient computation of the gradients for differentiable transient rendering greatly improves the reconstruction speed compared to other rendering based NLoS reconstructions. Our implementation is general enough to handle many cases and yields reconstructions quantitatively better than other approaches. Paired with a background network we were able to show results on a large range of simulated and real measurements. As the implementation is integrated into the PyTorch environment, it offers great flexibility and we have demonstrated its use in a self-supervised learning application. Furthermore, it may serve as a building block for future end-to-end training approaches or methods that also make use of the latest neural scene representations.

A major limitation of our method is its restriction to three-bounce, pulse-based setups, a necessity to achieve the highest possible performance for non-line-of-sight problems. As future work, we can imagine to extend the software by implementing gradients with respect to scan positions to allow for calibration similar to [17], but using more complex targets.

Acknowledgements. This work was supported by the European Research Council under ERC Starting Grant "ECHO" (802192).

References

[1] Nils Abramson. Light-in-flight recording by holography. *Optics letters*, 3(4):121–123, 1978.

[2] Victor Arellano, Diego Gutierrez, and Adrian Jarabo. Fast back-projection for non-line of sight reconstruction. *Optics express*, 25(10):11574–11583, 2017.

[3] Mauro Buttafava, Jessica Zeman, Alberto Tosi, Kevin Eliceiri, and Andreas Velten. Non-line-of-sight imaging using a time-gated single photon avalanche diode. *Optics express*, 23(16):20997–21011, 2015.

[4] Wenzheng Chen, Fangyin Wei, Kiriakos N Kutulakos, Szymon Rusinkiewicz, and Felix Heide. Learned feature embeddings for non-line-of-sight imaging and recognition. *ACM Transactions on Graphics (TOG)*, 39(6):1–18, 2020.

[5] Miguel Galindo, Julio Marco, Matthew O'Toole, Gordon Wetzstein, Diego Gutierrez, and Adrian Jarabo. A dataset for benchmarking time-resolved non-line-of-sight imaging, 2019.

[6] Javier Grau, Markus Plack, Patrick Haehn, Michael Weinmann, and Matthias Hullin. Occlusion fields: An implicit representation for non-line-of-sight surface reconstruction. *arXiv preprint arXiv:2203.08657*, 2022.

[7] Javier Grau Chopite, Matthias B Hullin, Michael Wand, and Julian Iseringhausen. Deep non-line-of-sight reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 960–969, 2020.

[8] Felix Heide, Matthew O'Toole, Kai Zang, David B Lindell, Steven Diamond, and Gordon Wetzstein. Non-line-of-sight imaging with partial occluders and surface normals. *ACM Transactions on Graphics (ToG)*, 38(3):1–10, 2019.

[9] Felix Heide, Lei Xiao, Wolfgang Heidrich, and Matthias B Hullin. Diffuse mirrors: 3d reconstruction from diffuse indirect illumination using inexpensive time-of-flight sensors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3222–3229, 2014.

[10] Quercus Hernandez, Diego Gutierrez, and Adrian Jarabo. A computational model of a single-photon avalanche diode sensor for transient imaging. *arXiv preprint arXiv:1703.02635*, 2017.

[11] Julian Iseringhausen and Matthias B Hullin. Non-line-of-sight reconstruction using efficient transient rendering. *ACM Transactions on Graphics (TOG)*, 39(1):1–14, 2020.

[12] Mariko Isogawa, Dorian Chan, Ye Yuan, Kris Kitani, and Matthew O'Toole. Efficient non-line-of-sight imaging from transient sinograms. In *European Conference on Computer Vision*, pages 193–208. Springer, 2020.

[13] Adrian Jarabo, Julio Marco, Adolfo Munoz, Raul Buisan, Wojciech Jarosz, and Diego Gutierrez. A framework for transient rendering. *ACM Transactions on Graphics (ToG)*, 33(6):1–10, 2014.

[14] A. Jarabo, B. Masia, J. Marco, and D. Gutierrez. Recent Advances in Transient Imaging: A Computer Graphics and Vision Perspective. *ArXiv e-prints*, Nov. 2016.

[15] Achuta Kadambi, Hang Zhao, Boxin Shi, and Ramesh Raskar. Occluded imaging with time-of-flight sensors. *ACM Transactions on Graphics (ToG)*, 35(2):1–12, 2016.

[16] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[17] Jonathan Klein, Martin Laurenzis, Matthias B Hullin, and Julian Iseringhausen. A calibration scheme for non-line-of-sight imaging setups. *Optics Express*, 28(19):28324–28342, 2020.

[18] Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. Differentiable monte carlo ray tracing through edge sampling. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, 37(6):222:1–222:11, 2018.

[19] David B Lindell, Gordon Wetzstein, and Vladlen Koltun. Acoustic non-line-of-sight imaging. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6780–6789, 2019.

[20] David B Lindell, Gordon Wetzstein, and Matthew O'Toole. Wave-based non-line-of-sight imaging using fast fk migration. *ACM Transactions on Graphics (TOG)*, 38(4):1–13, 2019.

[21] Xiaochun Liu, Sebastian Bauer, and Andreas Velten. Analysis of feature visibility in non-line-of-sight measurements. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10140–10148, 2019.

[22] Xiaochun Liu, Ibón Guillén, Marco La Manna, Ji Hyun Nam, Syed Azer Reza, Toan Huu Le, Adrian Jarabo, Diego Gutierrez, and Andreas Velten. Non-line-of-sight imaging using phasor-field virtual wave optics. *Nature*, 572(7771):620–623, Aug. 2019.

[23] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM siggraph computer graphics*, 21(4):163–169, 1987.

[24] Julio Marco, Wojciech Jarosz, Diego Gutierrez, and Adrian Jarabo. Transient photon beams. In *ACM SIGGRAPH 2017 Posters*, pages 1–2. 2017.

[25] Fangzhou Mu, Sicheng Mo, Jiayong Peng, Xiaochun Liu, Ji Hyun Nam, Siddeshwar Raghavan, Andreas Velten, and Yin Li. Physics to the rescue: Deep non-line-of-sight reconstruction for high-speed imaging. *arXiv preprint arXiv:2205.01679*, 2022.

[26] Ji Hyun Nam, Eric Brandt, Sebastian Bauer, Xiaochun Liu, Marco Renna, Alberto Tosi, Eftychios Sifakis, and Andreas Velten. Low-latency time-of-flight non-line-of-sight imaging at 5 frames per second. *Nature communications*, 12(1):1–10, 2021.

[27] Matthew O'Toole, David B Lindell, and Gordon Wetzstein. Confocal non-line-of-sight imaging based on the light-cone transform. *Nature*, 555(7696):338–341, 2018.

[28] Adithya Kumar Pediredla, Mauro Buttafava, Alberto Tosi, Oliver Cossairt, and Ashok Veeraraghavan. Reconstructing rooms using photon echoes: A plane based model and reconstruction algorithm for looking around the corner. In *2017 IEEE International Conference on Computational Photography (ICCP)*, pages 1–12. IEEE, 2017.

[29] Leonid I Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268, 1992.

[30] Siyuan Shen, Zi Wang, Ping Liu, Zhengqing Pan, Ruiqian Li, Tian Gao, Shiying Li, and Jingyi Yu. Non-line-of-sight imaging via neural transient fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(7):2257–2268, 2021.

[31] Malcolm Slaney and Philip A Chou. Time of flight tracer. Technical report, Technical Report. Microsoft Research., 2014.

[32] Adam Smith, James Skorupski, and James Davis. Transient rendering. 2008.

[33] Chia-Yin Tsai, Aswin C Sankaranarayanan, and Ioannis Gkioulekas. Beyond volumetric albedo–a surface optimization framework for non-line-of-sight imaging. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1545–1555, 2019.

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[35] Andreas Velten, Thomas Willwacher, Otkrist Gupta, Ashok Veeraraghavan, Moungi G Bawendi, and Ramesh Raskar. Recovering three-dimensional shape around a corner using ultrafast time-of-flight imaging. *Nature communications*, 3(1):1–8, 2012.

[36] Bin Wang, Ming-Yang Zheng, Jin-Jian Han, Xin Huang, Xiu-Ping Xie, Feihu Xu, Qiang Zhang, and Jian-Wei Pan. Non-line-of-sight imaging with picosecond temporal resolution. *Physical Review Letters*, 127(5):053602, 2021.

[37] Cheng Wu, Jianjiang Liu, Xin Huang, Zheng-Ping Li, Chao Yu, Jun-Tian Ye, Jun Zhang, Qiang Zhang, Xiankang Dou, Vivek K Goyal, et al. Non–line-of-sight imaging over 1.43 km. *Proceedings of the National Academy of Sciences*, 118(10):e2024468118, 2021.

[38] Lifan Wu, Guangyan Cai, Ravi Ramamoorthi, and Shuang Zhao. Differentiable time-gated rendering. *ACM Transactions on Graphics (TOG)*, 40(6):1–16, 2021.

[39] Shumian Xin, Sotiris Nousias, Kiriakos N Kutulakos, Aswin C Sankaranarayanan, Srinivasa G Narasimhan, and Ioannis Gkioulekas. A theory of fermat paths for non-line-of-sight shape reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6800–6809, 2019.

[40] Shinyoung Yi, Donggun Kim, Kiseok Choi, Adrian Jarabo, Diego Gutierrez, and Min H Kim. Differentiable transient rendering. *ACM Transactions on Graphics (TOG)*, 40(6):1–11, 2021.

[41] Sean I. Young, David B. Lindell, Bernd Girod, David Taubman, and Gordon Wetzstein. Non-line-of-sight surface reconstruction using the directional light-cone transform. In *Proc. CVPR*, 2020.

[42] Cheng Zhang, Bailey Miller, Kan Yan, Ioannis Gkioulekas, and Shuang Zhao. Path-space differentiable rendering. *ACM transactions on graphics*, 39(4), 2020.