Non-Line-of-Sight Reconstruction using Efficient Transient Rendering

JULIAN ISERINGHAUSEN and MATTHIAS B. HULLIN, University of Bonn, Germany

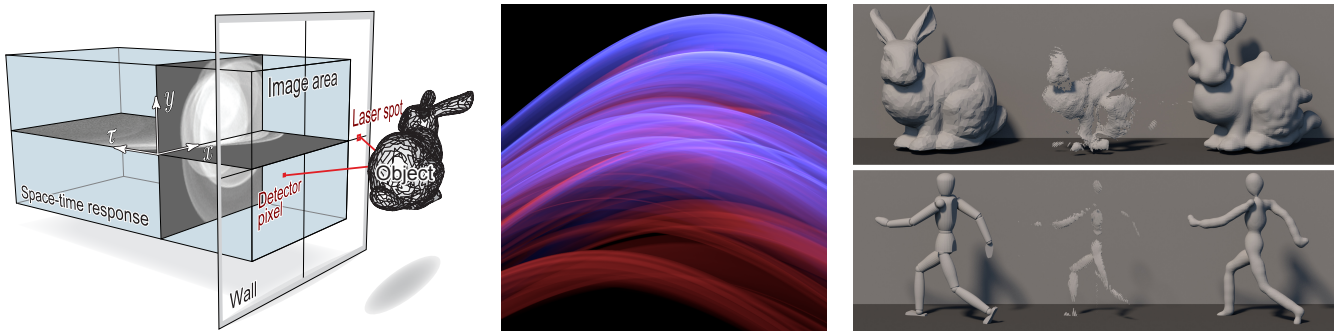

Fig. 1. (a) The challenge of looking around the corner deals with the recovery of information about objects beyond the direct line of sight. In this illustration of a setting proposed by Velten et al. [2012], an unknown object is located in front of a wall, but additional obstacles occlude the object from any optical devices like light sources or cameras. Our only source of information are therefore indirect reflections off other surfaces (here, a planar "wall"). A point on the wall that is illuminated by an ultrashort laser pulse turns into an omnidirectional source of indirect light ("laser spot"). After scattering off the unknown object, some of that light arrives back at the wall, where it forms an optical "echo" or space-time response (shown are 2D slices) that can be picked up by a suitable camera. Locations on the wall can be interpreted as omnidirectional detector pixels that receive different mixtures of backscattered light contributions at different times. We assume that neither camera nor laser can directly illuminate or observe the object, leaving us with the indirect optical space-time response as the only source of information. Note that for the sake of clarity, laser source, camera, and occluder are not shown here. The complete setup is illustrated in Figure 2. (b) We propose a novel transient renderer to simulate such indirectly scattered light transport efficiently enough for use as a forward model in inverse problems. In this artistic visualization, light contributions removed by the shadow test are marked in red, and the net intensity in blue. Together with an optimization algorithm, the renderer can be used to reconstruct the geometry of objects outside the line of sight. (c) Left to right: ground-truth object geometry; reconstruction using a state-of-the-art method (ellipsoidal backprojection); reconstruction using the technique presented in this paper. Top row: `BunnyGI` dataset; bottom row: `Mannequin1Laser` dataset. Our method relies on highly efficient and near-physical forward simulation, and it exemplifies the use of computer graphics as a technical tool to solve inverse problems in other fields.

Being able to see beyond the direct line of sight is an intriguing prospective and could benefit a wide variety of important applications. Recent work has demonstrated that time-resolved measurements of indirect diffuse light contain valuable information for reconstructing shape and reflectance properties of objects located around a corner. In this paper, we introduce a novel reconstruction scheme that, by design, produces solutions that are consistent with state-of-the-art physically-based rendering. Our method combines an efficient forward model (a custom renderer for time-resolved three-bounce indirect light transport) with an optimization framework to reconstruct object geometry in an analysis-by-synthesis sense. We evaluate our algorithm on a variety of synthetic and experimental input data, and show that it gracefully handles uncooperative scenes with high levels of noise or non-diffuse material reflectance.

Additional Key Words and Phrases: plenoptic imaging, inverse light transport, analysis by synthesis

This work was supported by the German Research Foundation (DFG) under grant (HU-2273/2-1) and the X-Rite Chair for Digital Material Appearance. We thank Hendrik Lensch, Michael Wand, Johannes Hanika, Felix Heide, Wolfgang Heidrich, Ivo Ihrke and Ramesh Raskar for valuable discussion over the course of this project. We also gratefully acknowledge the support of NVIDIA Corporation with the donation a Titan Xp GPU that was used in this research.

Authors' address: Julian Iseringhausen, iseringhausen@cs.uni-bonn.de; Matthias B. Hullin, hullin@cs.uni-bonn.de, University of Bonn, Institute of Computer Science II, Bonn, 53115, Germany.

1 MOTIVATION

Every imaging modality from ultrasound to x-ray knows situations where the target is partially or entirely occluded by other objects and therefore cannot be directly observed. In a recent strand of work, researchers have aimed to overcome this limitation, developing a variety of approaches to extend the line of sight of imaging systems, for instance using wave optics [Boger-Lombard and Katz 2018; Katz et al. 2014] or by using the occluder itself as an accidental imager [Bouman et al. 2017]. Among all the techniques proposed, a class of methods has received particular attention within the computer vision and imaging communities. The main source of information for these methods are indirect reflections of light within the scene, represented by time-resolved impulse responses. From such responses, it has been shown that the presence and position of objects "around a corner" [Kirmani et al. 2009], or even their shape [Velten et al. 2012] and/or reflectance [Naik et al. 2011] can be reconstructed. In this paper, we focus on the archetypal challenge of reconstructing the shape of an unknown object from 3-bounce indirect and (more or less) diffuse reflections off a planar wall (Figure 1(a)) [Kirmani et al. 2009]. The overwhelming majority of approaches to this class of problem rely on ellipsoidal *backprojection*, where intensity measurements are smeared out over the loci in space (ellipsoidal shells) that correspond to plausible scattering locations under the given

geometric constraints [Arellano et al. 2017; Buttafava et al. 2015; Gariepy et al. 2016; Kadambi et al. 2016; Velten et al. 2012]. Ellipsoidal backprojection implicitly assumes that the object is a volumetric scatterer, and it does not take into account surface orientation and self-occlusion of the object. More importantly, unlike linear backprojection used in standard emission or absorption tomography, ellipsoidal backprojection is not the adjoint of a physically motivated forward light transport operator. This necessitates heavy heuristic filtering [Manna et al. 2018], and the reconstructed shapes are typically flat and low in detail. On the other hand, algorithms based on ellipsoidal backprojection generally have much shorter runtimes than our approach, since they don't require a global optimization scheme.

Here, we propose an alternative approach that mitigates some of the problems of backprojection by formulating the non-line-of-sight sensing problem in an analysis-by-synthesis sense. In other words, we develop a physically plausible and efficient forward simulation of light transport (transient renderer) and combine it with a nonlinear optimizer to determine the scene hypothesis that best agrees with the observed data. The method is enabled by a number of technical innovations, which we consider the key contributions of this work:

- a scene representation based on level sets and a surface-oriented scattering model for time-resolved light transport around a corner (wall to object to wall) based on time-resolved radiative transfer,
- an extremely efficient GPU-based custom renderer for three-bounce backscatter that features near-physical handling of occlusion effects and a novel temporal filtering scheme for triangular surfaces, and
- a global, self-refining optimization strategy to minimize the reconstruction error.

We evaluate our method on a number of synthetic and experimental datasets and find that it is capable of achieving significantly higher object coverage and detail than ellipsoidal backprojection, even on greatly reduced and degraded input data. Our renderer not only naturally accommodates surface BRDFs, but is also open to extensions like higher-order light bounces or advanced background models that will be needed in order to tackle future non-line-of-sight sensing problems. The method, as proposed here, is not capable of delivering high reconstruction rates in this first implementation. However, we believe that being able to generate transient renderings for the around-the-corner setting very efficiently will enable novel approaches to the problem, for instance based on machine learning.

2 RELATED WORK

The research areas of transient imaging and non-line-of-sight reconstruction have recently received tremendous attention from the computer vision, graphics, imaging and optics communities. For a structured overview on the state of the art, we refer the interested reader to a recent survey [Jarabo et al. 2017].

2.1 Transient imaging

Imaging light itself as it propagates through space and time poses the ultimate challenge to any imaging system. To obtain an idea of the frame rate required, consider that in vacuum, light only takes about 3 picoseconds ($3 \cdot 10^{-12}\ s$) per millimeter of distance traversed. The typical transient imaging system consists of an ultrashort (typically, sub-picosecond) light source and an ultrafast detector. Oddly, three of the highest-performing detection technologies are over 40 years old: streak tubes [Velten et al. 2011] wherein a single image scanline is "smeared out" over time on a phosphor screen; holography using ultrashort pulses [Abramson 1978], and gated image intensifiers [Laurenzis and Velten 2014]. More common nowadays, however, are semiconductor devices that achieve comparable temporal resolution without the need for extreme light intensities or voltages. Among the technologies reported in literature are regular reverse-biased photodiodes [Kirmani et al. 2009], as well as time-correlated single-photon counters which conveniently map to standard CMOS technology [Gariepy et al. 2015]. On the low end, it has also been shown that transient images can be computationally reconstructed from multi-frequency correlation time-of-flight measurements [Heide et al. 2013], although data thus obtained typically suffers from the low temporal bandwidth of these devices, which necessitates heavy regularization.

2.2 Transient rendering

The simulation of transient light transport, when done naïvely, is no different from regular physically-based rendering, except that for each light path that contributes to the image, its optical length must be calculated and its contribution stored in a time-of-flight histogram [Smith et al. 2008]. A number of offline transient renderers have been made available to the public [Jarabo et al. 2014; Slaney and Chou 2014]. Even with advanced temporal sampling [Jarabo et al. 2014] and efficiency-increasing filtering strategies such as photon beams [Marco et al. 2017], such renderers still take on the order of hours to days to produce converged results. In contrast, the special-purpose renderer introduced in this paper is capable of producing close-to-physical renderings of around-the-corner settings in a matter of milliseconds. Finally, there have been efforts to simulate the particular characteristics of single-photon counters [Hernandez et al. 2017], an emerging type of sensor that can be expected to assume a major role in transient imaging.

2.3 Analysis of transient light transport and looking around corners

The information carried by transient images has been the subject of several investigations. Wu et al. laid out the geometry of space-time streak images for lensless imaging [2012], and discussed the influence of light transport phenomena such as subsurface scattering on the shape of the temporal response [2014]. Economically, the most important use of transient light transport analysis today is likely in multi-path backscatter removal for correlation-based time-of-flight ranging [Fuchs 2010, and many others].

In this paper, however, we direct our main attention to the idea of exploiting time-resolved measurements of indirect reflections for the purpose of extending the direct line of sight and, in effect, looking around corners [Kirmani et al. 2009; Velten et al. 2012]. While a variety of geometric settings have been investigated, the bulk of work in this area relies on the arrangement illustrated in

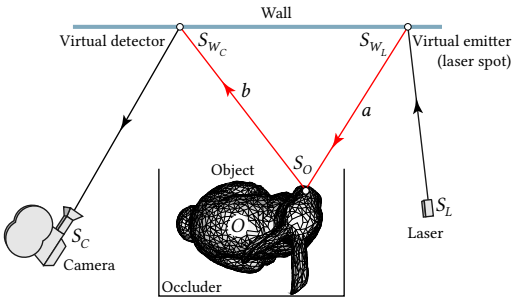

Fig. 2. Schematic top view of the scene arrangement, where the unknown object is occluded from direct observation. We assume that the temporal response has been "unwarped" (e.g., [Kadambi et al. 2016]), so only the occluded segments a and b contribute to the total time of flight and to the shading in Equation 4.

Figure 1(c) and Figure 2 and further introduced in the following Section 3. The reconstruction strategies can be roughly grouped in two classes. One major group is formed by backprojection approaches where each input measurement casts votes on those locations in the scene where the light could have been scattered [Arellano et al. 2017; Buttafava et al. 2015; Gariepy et al. 2016; Kadambi et al. 2016; Laurenzis and Velten 2014; Velten et al. 2012]. A smaller but more diverse group of work relies on the use of forward models to arrive at a scene hypothesis that best agrees with the measured data. Here, reported approaches range from combinatorial labeling schemes [Kirmani et al. 2009] via frequency-domain inverse filtering (if the capture geometry is sufficiently constrained) [O'Toole et al. 2018a] to variational methods using simple linearized light transport tensors [Heide et al. 2014; Naik et al. 2011] and simplistic models based on radiative transfer [Klein et al. 2016; Pediredla et al. 2017] that are (in principle) capable of expressing opacity effects like shadowing and occlusion, and physically plausible shading and that are closest to our proposed method. In concurrent work, Heide et al. [2017] added such extra factors as additional weights into their least-squares data term, achieving non-line-of-sight reconstructions of significantly improved robustness. Thrampoulidis et al. [2017] applied a similar idea on the reconstruction of 2D albedo maps on known geometry that are further obscured by known occluders between object and wall. With the proposed method, we demonstrate what we believe is the first reconstruction scheme for non-line-of-sight object geometry that is based on a near-physical yet extremely efficient special-purpose renderer and, by design, produces solutions that are self-consistent. We believe that our work can serve as an example for other uses of computer graphics methodology as a technical tool for solving inverse problems in imaging and vision.

3 PROBLEM STATEMENT

Here we introduce the geometry of the non-line-of-sight reconstruction problem as used in the remainder of the paper. For simplicity, we neglect the constant factor c (the speed of light) connecting *time* and *(optical) path length*. Thus, time and distance can be used synonymously and all discussions become independent of the absolute scale.

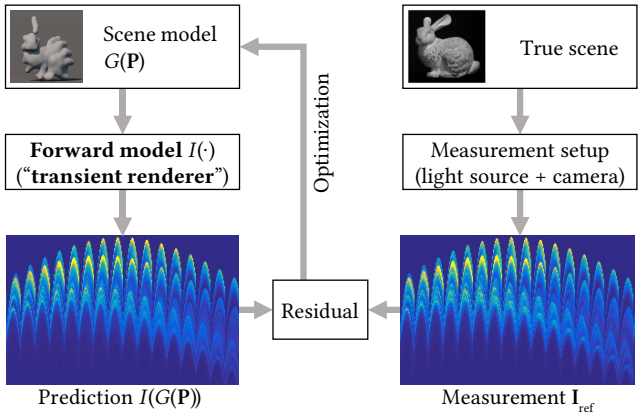

Fig. 3. Overview of our analysis-by-synthesis scheme for looking around a corner. Our pipeline heavily relies on custom-made components (scene representation, renderer, residual function, optimizer) to make this approach viable.

3.1 Problem geometry and transient images

We model our setting after the most common scenario from literature (Figure 2), where the unknown object is observed indirectly by illuminating a wall with a laser beam and measuring light reflected back to the wall. Following Kadambi et al. [2016], the laser spot on the wall acts as an area light source, and observed locations on the wall are equivalent to omnidirectional detectors that produce an "unwarped" transient image [Velten et al. 2013] (Figure 1(a)). The extent of the observed wall, the size of the object and its distance to the wall are usually on the same order of magnitude. The *transient image* or *space-time response* $\mathbf{I} \in \mathbb{R}^{n_x \times n_\tau}$ is the entirety of measurements taken using this setup, n_x being the number of combinations of detector pixels and illuminated spots and n_τ the number of bins in a time-of-flight histogram recorded per location. For a two-dimensional array of observed locations (for instance, when using a time-gated imager), the space-time response can be interpreted as a three-dimensional data cube similar to a video.

3.2 Problem formulation

The idea underlying ellipsoidal backprojection is that any entry in the transient image, or the response of a pair of emitter and detector positions for a given travel time, corresponds to an ellipsoidal locus of possible candidate scattering locations. If no further information is available, any measured quantity of light therefore "votes" for all locations on its ellipsoid. Finally, the sum or product of all such votes is interpreted as occupancy measure, or probability of there being an object at any point in space. We refer to a recent study [Manna et al. 2018] that discusses the design options for such algorithms in great detail.

In contrast, we formulate the reconstruction task as a non-linear least-squares minimization problem

$$\min_{\mathbf{P}} \|\mathbf{I}_{\text{ref}} - I\left(G\left(\mathbf{P}\right)\right)\|_2^2, \tag{1}$$

where $\mathbf{P}$ is a parameter vector describing the scene geometry, $G(\cdot)$ is a function that generates explicit scene geometry (a triangle

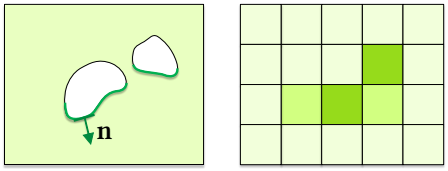

Fig. 4. Our light transport model relies on a surface-based object representation (left) with well-defined, opaque, oriented primitives (with normal vector **n**) that allow for shading and visibility tests including self-occlusion. Most other reconstruction techniques such as backprojection or O'Toole's method [2018a], on the other hand, use a volumetric representation (right) where each voxel carries a scattering density but no information on surface localization and orientation.

mesh), $\mathbf{I}_{\text{ref}}$ is the measured space-time scene response, and $I(\cdot)$ is a forward model (renderer) that predicts the response under the scene hypothesis passed as argument. The purpose of the optimization is to find the scene geometry $G(\mathbf{P})$ that minimizes the sum of squared pixel differences between the predicted and the observed responses. Figure 3 illustrates this principle.

A key feature of this formulation is that the solution by its very definition is optimally consistent with the chosen physical model of light transport, and that ongoing improvements in forward modeling will also benefit the reconstruction. Furthermore, our approach naturally handles opaque, oriented surfaces, whereas in backprojection, surface geometry is implicitly defined and needs to be derived using additional filtering steps (Figure 4). Furthermore, our method is able to handle arbitrary surface BRDFs, where current backprojection methods implicitly assume diffuse BRDFs. A downside of our approach is that it requires a full model of the scene, and that any unknowns (such as background or noise) can distort the solution in ways that are hard to predict. On the other hand, we believe that our approach lends itself for future extensions like higher-order light bounces.

4 METHOD

In the following, we introduce the components of our reconstruction algorithm in detail.

4.1 Geometry representation

We seek to parameterize the scene geometry in terms of a vector $\mathbf{P}$ that has a small number of degrees of freedom to make the optimization in Equation 1 tractable. Rather than using $\mathbf{P}$ to directly store a mesh representation with vertices and faces, we express the geometry implicitly as an isosurface of a scalar field $M_{\mathbf{P}}(\mathbf{x})$ composed of globally supported basis functions. This approach is also common in surface reconstruction from point clouds [Carr et al. 2001]. In our case, the vector $\mathbf{P}$,

$$\begin{aligned}\mathbf{P} &= (\mathbf{p}_1, \ldots, \mathbf{p}_m) \\ &= ((\mathbf{x}_1, \sigma_1), \ldots, (\mathbf{x}_m, \sigma_m)),\end{aligned} \tag{2}$$

lists the centers $\mathbf{x}_i$ and standard deviations σ_i of m isotropic Gaussian blobs. From the scalar field

$$M_{\mathbf{P}}(\mathbf{x}) = \sum_{i=1}^{m} e^{-\|\mathbf{x}-\mathbf{x}_i\|_2^2/(2\sigma_i^2)} \tag{3}$$

we extract the triangle mesh $G(\mathbf{P})$ using a GPU implementation of Marching Cubes [Lorensen and Cline 1987]. For all our reconstructions, we used a fixed resolution of 128^3 voxels for the reconstruction volume, and a fixed threshold of $\frac{3}{4}$ for the isosurface. The extension to other implicit functions, such as anisotropic Gaussians or general radial basis functions, is trivial.

4.2 Rendering (synthesis)

We propose a custom renderer that is suitable for use as forward model $I(\cdot)$ inside the objective function, Equation 1. In order to be suited for this purpose, the renderer must be sufficiently close to physical reality. At the same time, it has to be very efficient because hundreds of thousands of renderings may be required over the course of the optimization run. We achieve this efficiency by restricting the renderer to a single type of light path and rendering only light bounces from the wall to the object and back to the wall. Following the notation of [Pharr and Humphreys 2010] and by dropping any constant terms, we can write the incoming radiance for each camera pixel as

$$L = \int_O f\left(S_{W_L} \to S_O \to S_{W_C}\right) \eta\left(S_O \leftrightarrow S_{W_C}\right) \eta\left(S_{W_L} \leftrightarrow S_O\right) \mathrm{d}S_O, \tag{4}$$

where $O = G(\mathbf{P})$ denotes the object, f the object's BRDF and $S_{\_}$ surface points as shown in Figure 2. The geometric coupling term η is defined as

$$\eta\left(S_1 \leftrightarrow S_2\right) = V\left(S_1 \leftrightarrow S_2\right) \frac{|\cos(\theta_1)|\,|\cos(\theta_2)|}{\|S_1 - S_2\|_2^2}, \tag{5}$$

with V being the binary visibility function and θ_i the angle of the ray connecting S_1 and S_2 to the respective surface normal. Since our object is already represented as a triangle mesh, we are able to approximate Equation 4 by assuming a constant radiance over each triangles' surface,

$$\begin{aligned}L &\approx \sum_{t \in T} f\left(S_{W_L} \to S_t \to S_{W_C}\right) \eta\left(S_t \leftrightarrow S_{W_C}\right) \eta\left(S_{W_L} \leftrightarrow S_t\right) A_t \\ &=: \sum_{t \in T} \alpha_t .\end{aligned} \tag{6}$$

Here, T is the set of all triangles of our object, P_t is the centroid, and A_t the area. We denote the total irradiance contributed by triangle t as α_t. In our experiments, we use Lambertian and metal BRDFs, but other reflectance functions can be used as well. This approximation can be seen as an extension of the one found in [Klein et al. 2016]. We further add two important features to increase physical realism and generate a smooth transient image.

Our first addition are visibility tests (V) for both segments of the light path, which is necessary for handling non-convex objects. We first connect the laser point and the triangle centroid by a straight line, and test whether this segment intersects with any of the other triangles of the object mesh. For all visible triangles for which no intersection is found, we test the visibility of the second path segment (return of scattered light to the wall) in the same way. This shadow

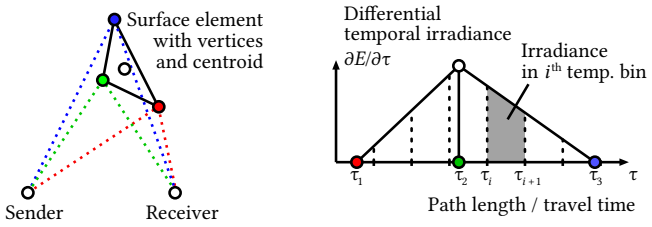

Fig. 5. To compute the total irradiance α_t contributed by a surface triangle to a given detector pixel, we evaluate the radiative transfer using the element's centroid. We then use a first-order filter to distribute this irradiance over the temporal bins that are affected by the triangle. To this end, we compute the three optical path lengths, or travel times, $\tau_{1\ldots3}$ belonging to the triangle's three vertices. The irradiance ending up in any temporal bin is then obtained by constructing a triangular function of total area α_t using the three arrival times as illustrated, then geometrically integrating over the time interval that corresponds to the bin.

test avoids overestimation of backscatter from self-occluding object surfaces. We note, however, that our way of performing the test only for the triangle centroid leads to a binary decision (triangle entirely visible or entirely shadowed) and therefore potentially makes the objective non-continuous. This can be reduced by using a triangle grid of sufficiently high resolution.

To render a transient image, we extend the pixels of the steady-state renderer to record time-of-flight histograms. The light contribution α_t enters into this histogram according to the geometric length of the corresponding light path; this length is simply the sum of the two Euclidean distances from laser point to point on triangle and back to the receiving point on the wall (see Figure 2). We found that the temporal response is prone to artifacts if only the centroid of the triangle is taken into account for the path length. Instead, we use the path lengths for the triangle's three corner vertices to determine the temporal footprint of the surface element. Using a linear filter, we then distribute the contribution α_t over the temporal domain (Figure 5). This procedure ensures that the rendered outcome is smooth in the temporal and spatial domains even when a single surface element covers dozens of temporal bins (Figure 6).

4.3 Optimization (analysis)

The optimization problem in Equation 1 is non-convex and non-linear, so special care has to be taken to find a solution (a set of blobs) that, when rendered, minimizes the cost function globally. While it would be desirable to optimize over the whole parameter vector $\mathbf{P}$ simultaneously, this is computationally prohibitive. To address this problem, we developed the iterative optimization scheme summarized in Algorithm 1, with subroutines provided in Algorithms 2 and 3.

The heart of our optimization algorithm is the inner optimization loop ITERATE($\mathbf{p}$, $\mathbf{P}$), which determines the $k = 10$ nearest neighbors of a given pivot blob $\mathbf{p}$ using the routine FIND_NEIGHBORS($\mathbf{p}$, $\mathbf{P}$). It then optimizes the *positions* of those blobs using the Levenberg-Marquardt algorithm, LEVENBERG_MARQUARDT($\mathbf{P}$) [Levenberg 1944; Marquardt 1963]. The function SET_VARIABLE($\mathbf{x}$) is used to label these parameters as variable to the solver, while all other blobs are

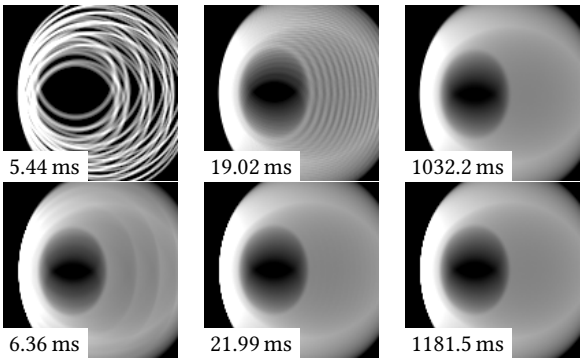

Fig. 6. The temporal filter also results in overall smoother spatial slices of the space-time response. Here we verify the performance of the filter by rendering the response generated by a planar square using different levels of detail. Shown is a single time slice without (top row) and with temporal filtering (bottom row). From left to right: coarse tesselation (4×4 quads), medium tesselation (16×16 quads), fine tesselation (128×128 quads). Numbers indicate the rendering time for the entire transient data cube (128×128 pixels, 192 time bins) on an NVIDIA GTX 980. Note the significant quality improvement at only 14–17% increased computational cost.

Algorithm 1 Global optimization scheme

Input: Reference image $\mathbf{I}_{\text{ref}}$, Threshold c_{thresh}
Output: Parameter vector $\mathbf{P}$, Cost c

$\mathbf{x} \leftarrow$ SAMPLE($\varnothing$)
$\mathbf{P}, c \leftarrow$ ADD_BLOB($\varnothing$, $\mathbf{x}$)
while $c > c_{\text{thresh}}$ **do**
  $\mathbf{x} \leftarrow$ SAMPLE($\mathbf{P}$)
  $\mathbf{P}_1, c_1 \leftarrow$ ADD_BLOB($\mathbf{P}$, $\mathbf{x}$)
  $\mathbf{P}_2, c_2 \leftarrow$ DUPLICATE_BLOB($\mathbf{P}$, $\mathbf{x}$)
  $\mathbf{P}_3, c_3 \leftarrow$ DELETE_BLOB($\mathbf{P}$, $\mathbf{x}$)
  $i \leftarrow \arg\min_x c_x$
  if $c_i < c$ **then**
    $\mathbf{P}, c \leftarrow \mathbf{P}_i, c_i$
  $\mathbf{P}_r, c_r \leftarrow$ REITERATE($\mathbf{P}$)
  if $c_r < c$ **then**
    $\mathbf{P}, c \leftarrow \mathbf{P}_r, c_r$
  $\mathbf{P}, c \leftarrow$ CHECK_DELETE($\mathbf{P}, c$)

Algorithm 2 Inner optimization scheme

function ITERATE($\mathbf{p}$, $\mathbf{P}$)
  $\mathbf{P}_{\text{opt}} \leftarrow$ FIND_NEIGHBORS($\mathbf{p}$, $\mathbf{P}$, 10)
  SET_FIXED($\mathbf{P}$)
  for all $(\tilde{\mathbf{p}}, \tilde{\sigma}) \in \mathbf{P}_{\text{opt}}$ **do**
    SET_VARIABLE($\tilde{\mathbf{p}}$)
  $\mathbf{P} \leftarrow$ LEVENBERG_MARQUARDT($\mathbf{P}$)
  for all $(\tilde{\mathbf{p}}, \tilde{\sigma}) \in \mathbf{P}_{\text{opt}}$ **do**
    SET_VARIABLE($\tilde{\mathbf{p}}$)
    SET_VARIABLE($\tilde{\sigma}$)
  $\mathbf{P} \leftarrow$ LEVENBERG_MARQUARDT($\mathbf{P}$)
  $c \leftarrow$ COMPUTE_COST($\mathbf{P}$)
  return $\mathbf{P}$, c

kept fixed during the optimization run using SET_FIXED($\mathbf{x}$). Derivatives for the Jacobian matrix are computed numerically using finite differences (by repeatedly executing our forward renderer with the perturbed parameter vector). In a subsequent step, the sizes of the selected blobs are also included in a second optimization run, with a parameter $\sigma_{\max}$ defining an upper limit for the blob size. We found that this two-stage approach is necessitated by the strong non-convexity of the objective function. By optimizing over multiple blobs simultaneously, we allow the optimizer to recover complex geometry features that are influenced by more than a single blob.

The algorithm starts with a single blob as initial solution, then performs an outer loop over four phases: sampling, mutation, reiteration, and regularization. In the following, we provide a full description of the individual phases and explain our design choices. The parameters used in our reconstructions are shown in Table 2.

Sampling. Our algorithm pivots around locations in the reconstruction volume that are chosen according to a distribution (PDF) that aims to give problematic regions a higher probability of being sampled. We obtain the PDF by backprojecting the absolute value of the current residual image into the working volume. For locations $\mathbf{x}$ that are sampled by the function SAMPLE(), our working hypothesis is that *something* about the solution should change there; we address this by selecting the nearest blob to this location (FIND_NEAREST($\mathbf{P}$, $\mathbf{x}$)) and applying and testing our three mutation strategies on it. Since each mutation probably increases the cost function, it is followed by a relaxation of the neighborhood of the pivot blob.

Mutation. We employ three mutation strategies to generate variations of the current solution. ADD_BLOB($\mathbf{P}$, $\mathbf{x}$) adds a new blob $(\mathbf{x}, \sigma_0)$ to $\mathbf{P}$. DELETE_BLOB($\mathbf{P}$, $\mathbf{x}$) deletes the blob $\mathbf{p} \in \mathbf{P}$ that is closest to $\mathbf{x}$. DUPLICATE_BLOB($\mathbf{P}$, $\mathbf{x}$) replaces the blob $\mathbf{p} \in \mathbf{P}$ by two new blobs that are displaced by a vector $\pm\mathbf{d}$ from the original position so they can be separated by the optimizer. Out of the three solutions (each one after performing an inner optimization ITERATE($\mathbf{p}$, $\mathbf{P}$) on the neighborhood), the one with the lowest cost c_i is chosen to be the new solution.

Reiteration. As the next step, another call to ITERATE is performed on a random group of neighboring blobs. This re-evaluation of previously relaxed blobs is necessary to avoid being stuck in local minima during early iterations, when the hypothesis does not yet contain enough blobs to properly describe the transient response.

Regularization. Finally, the algorithm first checks each blob for its significance to the solution (CHECK_DELETE), and deletes it if doing so does not worsen the total cost by more than a small factor η. This regularizing step prevents the build-up of excess geometry in hidden regions that is not supported by the data. It is the only step that can lead to an increase in the cost c; all other heuristics ensure that the cost falls monotonically.

4.4 Implementation details

Our reconstruction software is written in C++. Geometry generation and rendering are implemented on the GPU, using NVIDIA CUDA and the Thrust parallel template library for the bulk of the tasks and the NVIDIA OptiX prime ray-tracing engine for the shadow tests. The optimization algorithm is implemented using the Ceres solver [Agarwal et al. 2015]. Intermediate results are visualized on-the-fly using the VTK library [Schroeder et al. 2006]. We used various workstations in our experiments, with Intel Core i7 CPUs and NVIDIA GeForce GPUs ranging from GTX 780 to Titan Xp.

Algorithm 3 Subroutines to Algorithm 1.

```
function ADD_BLOB(P, x)
    p ← (x, σ_0)
    return ITERATE(p, P ∪ p)

function CHECK_DELETE(P)
    for all p ∈ P do
        if COMPUTE_COST(P \ p) < η · c then
            P ← P \ p
    c ← COMPUTE_COST(P)
    return P, c

function DUPLICATE_BLOB(P, x)
    p ← FIND_NEAREST(P, x)
    p_1, p_2 ← SPLIT(p)
    return ITERATE(p, P \ p ∪ p_1 ∪ p_2)

function REITERATE(P)
    p ← CHOOSE_RANDOM(P)
    return ITERATE(p, P)

function REMOVE_BLOB(P, x)
    p ← FIND_NEAREST(P, x)
    return ITERATE(p, P \ p)
```

5 EVALUATION

In this section, we verify the correctness of our renderer, and use it to reconstruct geometry from simulations and experimental measurements of around-the-corner scattered light.

5.1 Correctness of renderer

Before we evaluate the performance of our overall reconstruction system, we test correctness and performance of the forward model that is at its heart, our custom renderer. To this end, we prepare test scenes and render reference images using Microsoft's Time of Flight Tracer [Slaney and Chou 2014], a transient renderer based on `pbrt` version 2 [Pharr and Humphreys 2010].

All our synthetic models use the same arbitrary unit for length and time. The standard temporal resolution (size of histogram bin) of our virtual detectors is 0.4 units. Typical time resolutions of real-world devices are 10 ps for streak cameras or 100 ps for SPAD detectors. Equating the bin size with these time constants results in a conversion factor to real-world distances of 8.3 mm and 83 mm per world unit, respectively. We arranged the scene such that the wall is a diffuse plane at $z = 0$ with normal in positive z direction. The object, with a typical size of 50 units, was located on the z axis at $z = 45$. The laser spot was modeled as a cosine-lobe light source pointing in positive z direction at one of four wall locations $(45, 0, 0)$, $(-45, 0, 0)$, $(0, 45, 0)$ and $(0, -45, 0)$. The range of observed points on

the wall was represented by an area of 80×80 units^2 which was observed by an orthographic camera centered at $(x, y) = (0, 0)$.

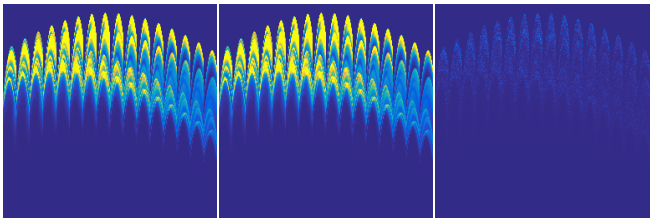

Fig. 7. The physically-based renderings with and without global illumination are virtually indistingishuable. From left to right: Rendering with global illumination; rendering without global illumination; difference of the two renderings.

Using a 30% reflective triangle mesh model of the Stanford Bunny, we generated two reference renderings of $16 \times 16 \times 256$ spatio-temporal resolution using the physically-based renderer, one with full global illumination and one with a maximum path length of 2 reflections. With the cosine light source representing the spot lit by the laser, a path length of 2 includes light scattering from the wall to the object and back to the wall, but not light thas has been interreflected at the object or that has bounced between object and wall multiple times. In Figure 7, both versions are shown along with the difference. At least for our around-the-corner setting, we found that the error caused by truncating the path length to 2 is not very significant, with 69.809 dB peak signal-to-noise ratio (PSNR) or a relative $L2$ difference of 0.486%.

We then used the truncated rendering as reference for our own renderer, and tested the effect of temporal filtering and shadow testing on the difference (Figure 8). A naïve version of our renderer, with all refinements disabled, reached the reference up to an error of a little under 10%. After activating the temporal filtering and the shadow tests, our fast renderer delivered a close approximation to the ray-traced reference with with 69.796 dB peak signal-to-noise ratio (PSNR) or a relative $L2$ difference of 0.489%.

All error values are provided at a glance in Table 1. The main result from this investigation is that both features are essential to our renderer. The gain in accuracy comes at the expense of significantly increased runtime when using the shadow test (Figure 10). For small numbers of pixels, a significant part of that runtime is caused by the construction of acceleration structures—here, about 10 ms for an object with approximately 55,000 triangles. Another noteworthy observation is that the Monte-Carlo rendering used as reference was likely not fully converged (Figure 9) even after evaluating 250 million samples per pixel. We expect that more exhaustive sampling would likely have further reduced the error.

5.2 Geometry reconstruction

We used various types of input data to test our algorithm: synthetic data generated using a path tracer or our own fast renderer, as well as experimental data obtained from other sources. The results from these reconstructions are scattered throughout the paper, referencing the datasets from Table 2 by their respective names. Meshes

Comparison		PSNR [dB]	Rel. $L2$ error [%]
RTFull	/ RTTrunc	69.809	0.486
RTTrunc	/ OursFull	69.796	0.489
RTTrunc	/ OursNoFilter	53.379	3.237
RTTrunc	/ OursNoShadow	45.638	7.892
RTTrunc	/ OursNoShadowNoFilter	44.942	8.550

Table 1. Using the Stanford Bunny as test object, we compare our renderer to ray-traced renderings with maximum path lengths of 2 (RTTrunc) and ∞ (RTFull). With all the features enabled (OursFull), our renderer matches the ray-traced solution for the 3-bounce setting (wall-object-wall) to 0.49%, which is on the same order as the influence of global illumination (RTFull) on this scene. Omission of shadow tests and temporal filtering result in significantly higher error values.

are rendered in a daylight environment using Mitsuba [Jakob 2010], with a back wall and ground plane added as shadow receivers for better visualization of the 3D shapes. Note that these planes are not part of the experimental setup.

Synthetic datasets. After establishing in Section 5.1 that our fast renderer produces outcomes that are almost identical to the ray-traced reference, we used both the path tracer and our fast renderer to generate a variety of around-the-corner input data. In particular, we prepared several variations of the `Mannequin` scene, reducing the number of pixels, the number of laser spots, as well as the temporal resolution. An overview of all our datasets, as well as the parameters used for reconstructing them, can be found in Table 2. Like the back-projection method, ours too has a small number of parameters: the upper bound for the blob size, σ_0 and the regularization parameter η.

We show renderings of the reconstructed meshes alongside the backprojected solutions, obtained using the Fast Backprojection code provided by Arellano et al. [2017], and ground truth (Figure 1(c)). They show that the quality delivered by our algorithm, in general, can compete with or even beats the state-of-the-art method. The meshes produced by our method tend to be more complete, smoother, and overall closer to the true surface. We also performed more quantitative evaluations. Figures 12 and 13 show the error of the recovered surface in z-direction. In general, meshes generated using the backprojection method tend to lie in front of the true surface. This is due to the way surface geometry is reconstructed from the density volumes obtained by the backprojection algorithm. Even if the peak of the density distribution lies exactly on the object geometry, extracting an isosurface will displace it by a certain distance. Our reconstructions, which are based on a surface scattering model, do not suffer from this effect.

Degradation experiments. To put the robustness of our method to the test, we performed a series of experiments that deliberately deviate from an idealized, noise-free, Lambertian and global-illumination-free light transport model, or reduce the amount of input data used for the reconstruction. In a first series of experiments, we sub-sampled the Mannequin dataset both spatially and temporally, and observed the degradation in reconstructed outcome (Figure 19). In a second series, we added increasing amounts of Poisson noise (Figure 20). Next, we replaced the diffuse reflectance of

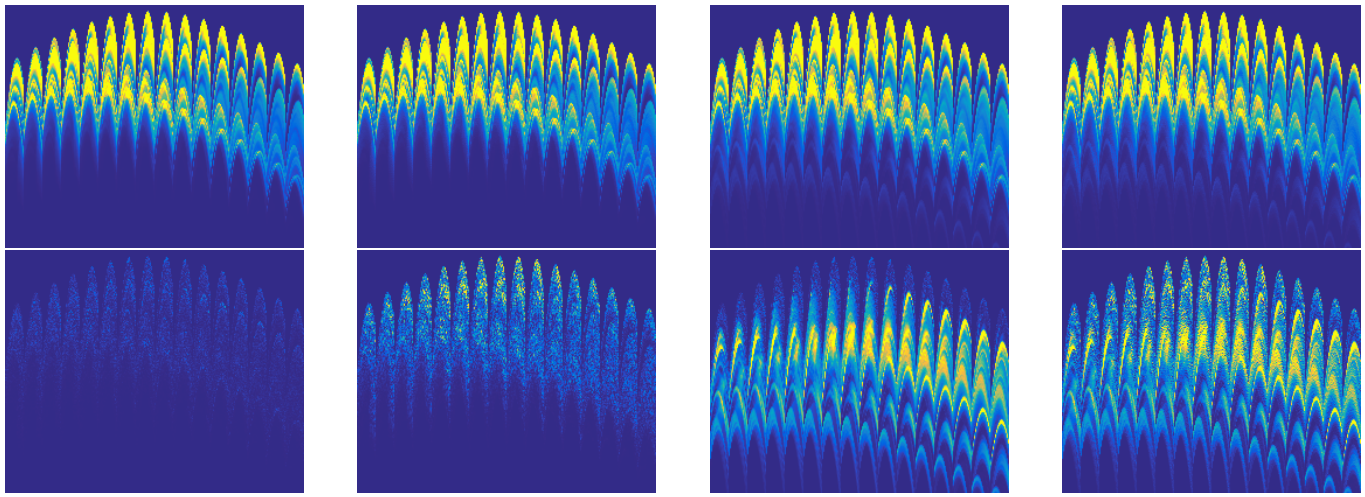

Fig. 8. The effect of our augmentations on the rendering error. The top row shows transient renderings made with our renderer, the bottom row shows the respective difference to the ground truth `toftracer` rendering (range scaled for print). From left to right: Our renderer with all features turned on; temporal filtering turned off; shadow tests turned off; temporal filtering and shadow tests turned off. Error metrics for these renderings are provided in Table 1.

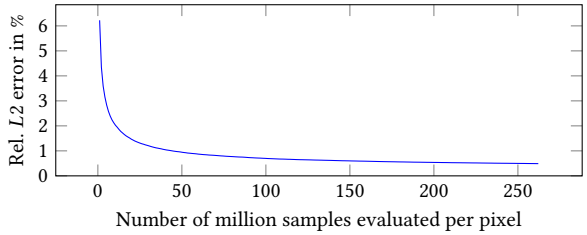

Fig. 9. Difference between our fast renderer and the ray-traced reference solution with a varying number of samples per pixels.

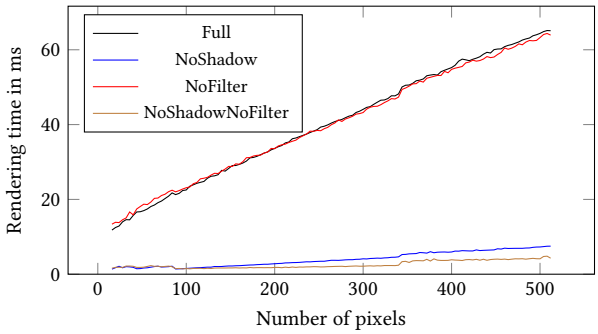

Fig. 10. Rendering performance of four versions of our algorithm (with/without filtering, with/without shadow test) as a function of output pixel count.

the `BunnyGI` model by a metal BRDF (Blinn model as implemented by `pbrt`) and decreased the roughness value (Figure 14). Our fast renderer used during reconstruction was set to the same BRDF parameters that were used to generate the input data. Finally, we constructed a strongly concave synthetic scene (`Bowl`) and used high albedo values in order to test the influence of unaccounted-for global illumination on the reconstructed geometry (Figure 15).

As expected, in all these examples, the further the data deviates from the ideal case, the more the reconstruction quality decreases. While backprojection tends to be more robust with respect to low-frequency bias (`Bowl` experiment), our method quite gracefully deals with high-frequency noise by fitting a low-frequent rendering to it. For highly specular materials, the discretization of the surface mesh and the sensing locations on the wall may lead to sampling issues: specular glints that are missed by the forward simulation cannot contribute to the solution.

Experimental datasets. We show reconstructions of two experimental datasets obtained using SPAD sensors.

The first dataset (`SPADScene`) was measured by Buttafava et al. [2015], by observing a single location on the wall with a SPAD detector, and scanning a pulsed laser to a rectangular grid of locations. We note that this setup is dual, and hence equivalent for our purpose, to illuminating the single spot and scanning the detector to the grid of different locations. The dataset came included with the Fast Backprojection code provided by Arellano et al. [2017]. To apply our algorithm on the `SPADScene` dataset, we first subtracted a lowpass-filtered version (with $\sigma = 1000$ bins) of the signal to reduce noise and background, then downsampled the dataset from its original temporal resolution by a factor of 25.

Like in the original work, the reconstruction remains vague and precise details are hard to make out (Figure 16). The reconstructed blobby objects appear to be in roughly the right places, but their shapes are poorly defined. We note that our method quite clearly carves out the letter "T" where backprojection delivers a less clearly defined shape (Figure 17).

The second dataset (`OTooleDiffuseS`) is a measurement of a letter "S" cut from white cardboard, which O'Toole et al. measured via a diffuse wall using their confocal setup [O'Toole et al. 2018a].

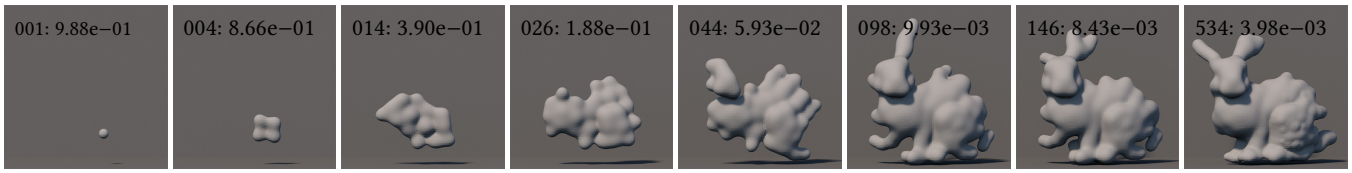

Fig. 11. Convergence of reconstructed geometry for the Bunny dataset over the course of the optimization. Number pairs denote iteration number and value of cost function (relative to start value).

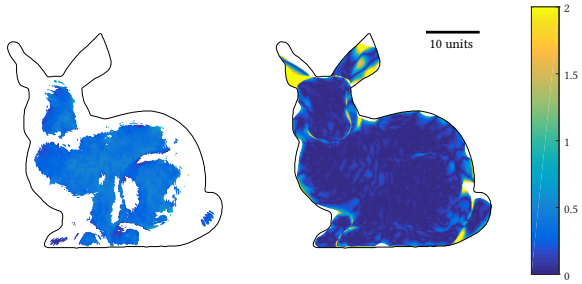

Fig. 12. Absolute depth error (in world units) in the reconstructions obtained from the synthetic Bunny dataset using four laser spot positions. Left: backprojection; right: ours. The black line indicates the ground-truth object silhouette.

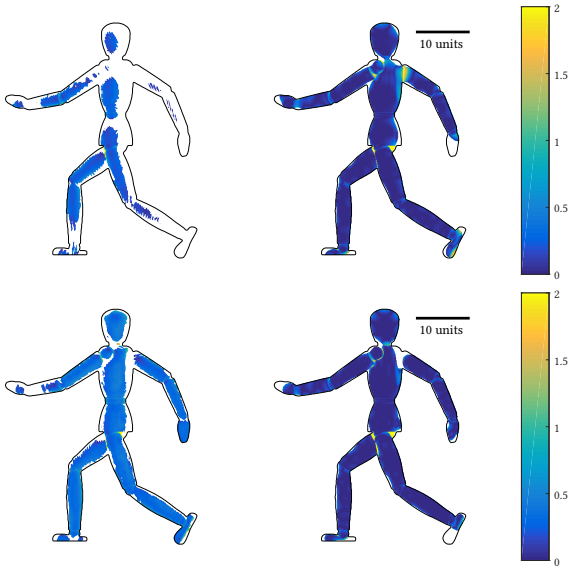

Fig. 13. Absolute depth error (in world units) in the reconstructions obtained from the synthetic `Mannequin` datasets using one (top) and four (bottom) laser spot positions. Left: backprojection; right: ours. The black line indicates the ground-truth object silhouette.

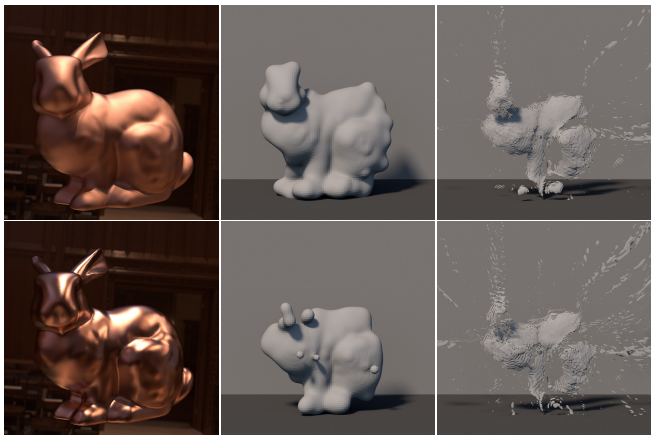

Fig. 14. Reconstruction of the `BunnyMetal*` scenes with `pbrt`'s `metal` BRDF applied to the object (top row: Blinn roughness 0.05; bottom row: Blinn roughness 0.01). From left to right: reference rendering in Grace Cathedral environment [Debevec 1998]; our proposed method; backprojection.

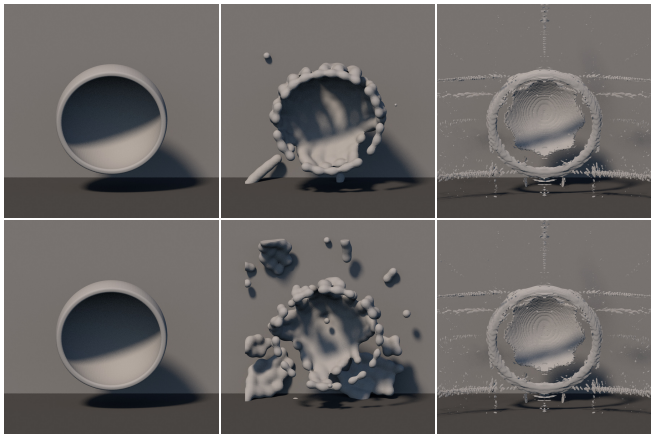

Fig. 15. `Bowl` scene. A strongly concave shape with high albedo (top row: 30%; bottom row: 100%) features large amounts of interreflected light in the input data, which leads to spurious features in the reconstructed geometry. From left to right: reference geometry; our proposed method; backprojection.

In this setup, illumination and observation share the same optical path and are scanned across the surface. We downsampled the input data by a factor of $4 \times 4 \times 4$ in the spatial and temporal domains. Although the inclusion of the direct reflection in the data allowed for a better background subtraction and white point correction than in the case of the previous dataset, it becomes clear that there must be more sources of bias. In particular, we identified a temporal blur of roughly 3 time bins. Adding a similar blur to our renderer (a box filter of width 3 bins), made the reconstructed "S" shape much more clearly recognizable as such (Figure 18).

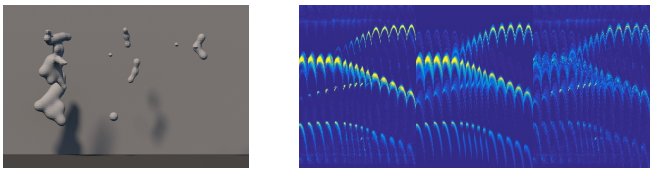

Fig. 16. Reconstruction of the experimental SPADScene dataset [Buttafava et al. 2015]. Shown is the output mesh and the transient data (from left to right: observation, prediction, residual).

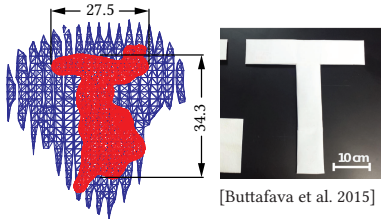

Fig. 17. The "T" object from the experimental SPADScene dataset published by Buttafava et al. [2015]. Shown are reconstructions obtained using back-projection (blue) and the proposed method (red), along with approximate dimensions using the scale provided in the original work (right).

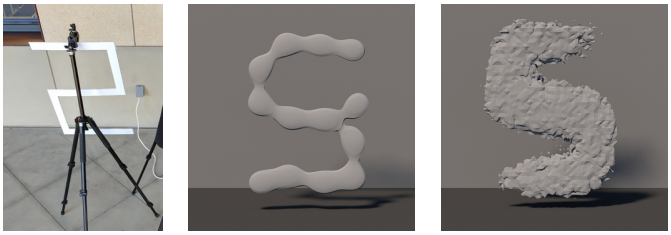

Fig. 18. OTooleDiffuseS dataset [O'Toole et al. 2018a]. From left to right: photo of diffuse "S"-shaped cutout; surface mesh reconstructed using our method; mesh reconstructed using method described in [O'Toole et al. 2018a].

6 DISCUSSION

In the proposed approach, we develop computer graphics methodology (a near-physical, extremely efficient rendering scheme) to reconstruct occluded 3D shape from three-bounce indirect reflections. To our knowledge, this marks the first instance of a non-line-of-sight reconstruction algorithm that is consistent with a physical forward model. This solid theoretical foundation leads to results that, under favorable conditions, show higher object coverage and detail than the de-facto state of the art, error backprojection. In extreme situations, like very low spatial / temporal resolutions or high noise levels, we have shown that our method breaks down significantly later than the current state of the art (Figures 19 and 20). Under conditions that are not covered by the forward model (noise, bias/background, global illumination) the results are on par or slightly inferior to existing methods. In terms of runtime, our method typically takes several hours or even days for a reconstruction run (Table 1) and therefore cannot compete with recent optimized versions of error backprojection [Arellano et al. 2017] or GPU-based deconvolvers [O'Toole et al. 2018b]. However, we consider this a soft hindrance that has to be considered together with the fact that the capture of suitable input data, too, is far from being instantaneous. This latter factor is governed by the physics of light and therefore may turn out, in the long run, to impose more severe limitations to the practicality of non-line-of-sight sensing solutions.

We noted that the reconstruction quality of the SPAD datasets stays behind the quality of the synthetic datasets (whether path-traced or using our own renderer). Our image formation model approximates the physical light transport up to very high accuracy (as shown in Section 5.1), but does not explicitly model the SPAD sensor response to the incoming light. The SPAD data is biased due to background noise and dark counts, and the temporal impulse response is asymmetric and smeared out due to time jitter and afterpulsing [Gulinatti et al. 2011; Hernandez et al. 2017]. While these effects could easily be incorporated into our forward model, doing so would require either a careful calibration of the imaging setup (which was not provided with the public datasets) or an estimation of the noise parameters from input data. In this light, we find the presented results very promising for this line of research, and consider the explicit application of measured noise profiles and the modeling of additional imaging setups as future work.

A key feature of our method is that, within the limitations of the forward model (opaque, but not necessarily diffuse, light transport without further interreflections) good solutions can be immediately identified by a low residual error. However, the non-convex objective and possibly unknown noise and background terms may make it challenging to reach this point. Our optimization scheme, while delivering good results in the provided examples, offers no guarantee of global convergence. As of today, it is unclear which of the two factors will prove more important in practice, the physical correctness of the forward model or the minimizability of the objective derived from it.

7 FUTURE WORK

We imagine that extended versions of our method could be used to jointly estimate geometry and material. Advanced global optimization heuristics could further improve the convergence behavior and the overall quality of the outcome. We imagine that hierarchical approaches or hybrid solutions might bring further improvement, for instance by using the (physically inaccurate but global) solution of one reconstruction scheme to warm-start another local optimization run using a more accurate model like ours.

Finally, our renderer is not constrained to use in a costly iterative solver. Just as well, we can imagine using it to enable new machine learning approaches to the problem. A suitably trained feedforward neural network, for example, would deliver instant results. Whereas existing renderers are too slow for generating large amounts of training data, our renderer would be fast enough to obtain millions of datasets in a single day. Together with a suitable signal degradation model [Hernandez et al. 2017], we expect that it will be possible to closely approximate the most relevant real-world scenarios.

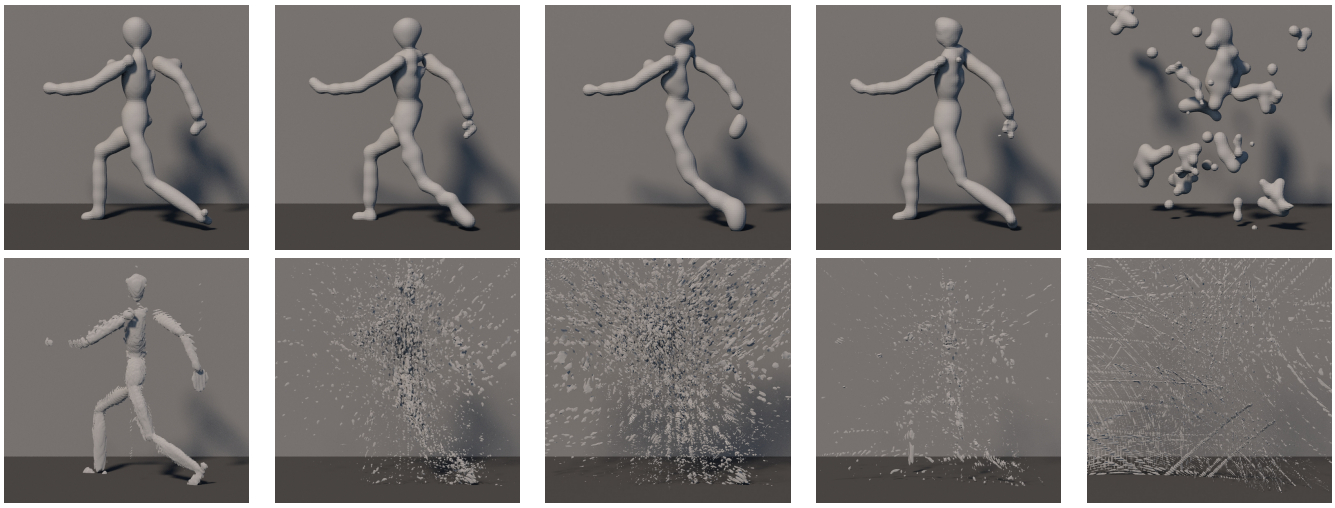

Fig. 19. Reconstruction of the `Mannequin*` dataset using different levels of degradation. From left to right: `Mannequin`, `MannequinLowTemp`, `MannequinMinTemp`, `MannequinLowRes`, `MannequinMinRes`. Top row: Our reconstruction, bottom row: backprojection. Unlike backprojection, our reconstruction method handles degradations in the input data quite gracefully. Even an extremely low spatial resolution of 2×2 pixels or a temporal resolution of only 8 bins still produces roughly identifiable results.

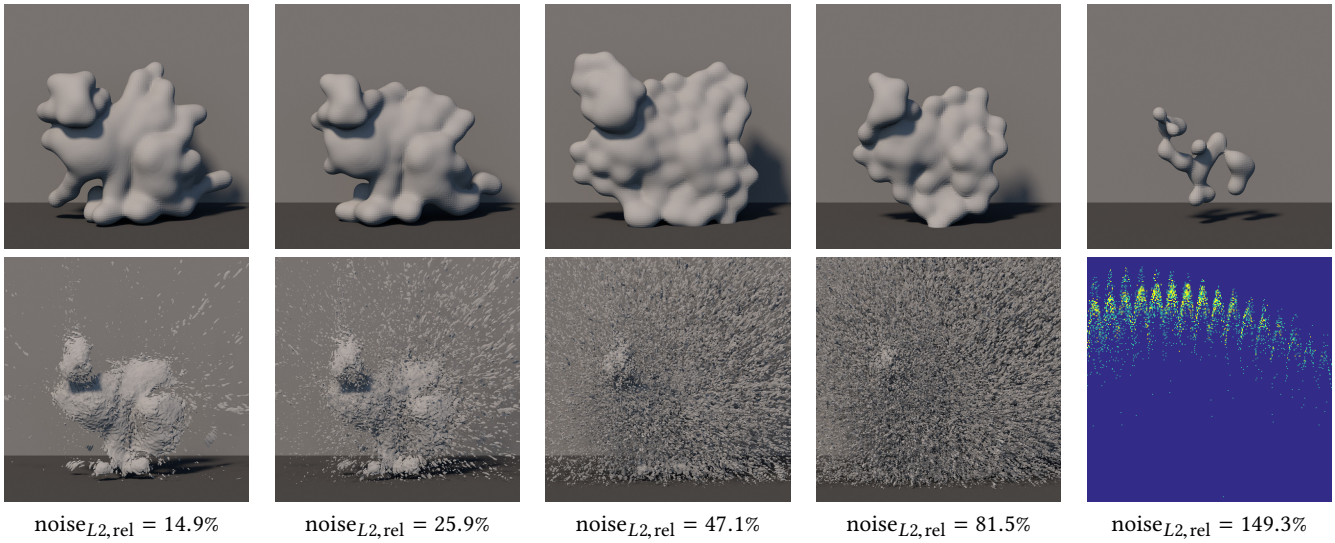

Fig. 20. Reconstruction of the `BunnyGI` dataset with different levels of Poisson noise applied to the input data. Relative $L2$ error from left to right: 14.9%, 25.9%, 47.1%, 81.5%, 149.3%. Top row: Our reconstruction, bottom row: backprojection. Our algorithm is based on a noise-free forward model. It therefore manages to localize the object reliably even under very noisy conditions (albeit at reduced reconstruction quality). In the rightmost example (streak plot), at most two photons have been counted per pixel, resulting in data that contains 50% more noise than signal.

REFERENCES

Abramson, N. 1978. Light-in-flight recording by holography. *Optics Letters* 3, 4 (Oct 1978), 121–123. https://doi.org/10.1364/OL.3.000121

Agarwal, S., K. Mierle, and Others [sic]. 2015. Ceres Solver. http://ceres-solver.org. (2015).

Arellano, V., D. Gutierrez, and A. Jarabo. 2017. Fast back-projection for non-line of sight reconstruction. *Optics Express* 25, 10 (2017).

Boger-Lombard, J. and O. Katz. 2018. Non line-of-sight localization by passive optical time-of-flight. *arXiv preprint arXiv:1808.01000* (2018).

Bouman, K. L., V. Ye, A. B. Yedidia, F. Durand, G. W. Wornell, A. Torralba, and W. T. Freeman. 2017. Turning Corners into Cameras: Principles and Methods. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2270–2278.

Buttafava, M., J. Zeman, A. Tosi, K. Eliceiri, and A. Velten. 2015. Non-line-of-sight imaging using a time-gated single photon avalanche diode. *Optics Express* 23, 16 (2015), 20997–21011.

Carr, J., R. Beatson, J. Cherrie, T. Mitchell, W. Fright, B. McCallum, and T. Evans. 2001. Reconstruction and representation of 3D objects with radial basis functions. In *Proc. 28th Annual Conf. on Computer Graphics and Interactive Techniques*. ACM, 67–76.

Name	Reference	Resolution	# Lasers	η	σ_0	t_0	δ_t	c_n/c_0 [%]	T [min]	n_{iters}	n_{blobs}
Bunny	Ours	$16 \times 16 \times 256$	4	1.01	1.5	80	0.4	0.32	5096	660	156
BunnyGI	pbrt	$16 \times 16 \times 256$	4	1.01	1.5	80	0.4	0.59	5611	181	109
BunnyMetal0.05	pbrt	$16 \times 16 \times 256$	4	1.01	1.5	80	0.4	2.02	3419	259	101
BunnyMetal0.01	pbrt	$16 \times 16 \times 256$	4	1.01	1.5	80	0.4	11.75	3005	361	87
BowlAlbedo0.3	pbrt	$16 \times 16 \times 256$	4	1.001	0.4	80	0.4	4.36	5579	167	155
BowlAlbedo1	pbrt	$16 \times 16 \times 256$	4	1.001	0.4	80	0.4	31.53	4280	267	197
Mannequin	Ours	$16 \times 16 \times 256$	4	1.005	1.5	90	0.4	1.464	2326	505	69
MannequinLowRes	Ours	$4 \times 4 \times 256$	4	1.005	1.5	90	0.4	1.414	1251	252	76
MannequinMinRes	Ours	$2 \times 2 \times 256$	4	1.005	1.5	90	0.4	4.444	931	350	101
MannequinLowTemp	Ours	$16 \times 16 \times 32$	4	1.005	1.5	90	3.2	1.208	1322	166	67
MannequinMinTemp	Ours	$16 \times 16 \times 8$	4	1.005	1.5	90	12.8	7.952	420	102	23
Mannequin1Laser	Ours	$16 \times 16 \times 256$	1	1.005	1.5	90	0.4	0.586	1419	243	57
SPADScene	Measured	$185 \times 1 \times 256$	1	1.005	4.5	373	0.748	20.313	1280	328	43
OTooleDiffuseS	Measured	$64 \times 64 \times 2048$	1	1.01	0.015	0.756	0.0012	33.431	67	13	13

Table 2. Parameters of our reconstructed scenes, temporal parameters, residual cost after optimization (relative to initial value), total runtime T of the reconstruction, number of iterations and number of blobs. All values for T are taken from file timestamps and vary due to manual termination of the reconstruction procedure, execution on different GPU models, overhead through parallel execution of multiple jobs, as well as debugging output.

Debevec, P. 1998. Light Probe Image Gallery. (1998). http://www.pauldebevec.com/Probes/.

Fuchs, S. 2010. Multipath interference compensation in time-of-flight camera images. In *Pattern Recognition (ICPR), 2010 20th International Conference on*. IEEE, 3583–3586.

Gariepy, G., N. Krstajic, R. Henderson, C. Li, R. Thomson, G. Buller, B. Heshmat, R. Raskar, J. Leach, and D. Faccio. 2015. Single-photon sensitive light-in-flight imaging. *Nature Communications* 6 (2015).

Gariepy, G., F. Tonolini, R. Henderson, J. Leach, and D. Faccio. 2016. Detection and tracking of moving objects hidden from view. *Nature Photonics* 10, 1 (2016).

Gulinatti, A., I. Rech, M. Assanelli, M. Ghioni, and S. Cova. 2011. A physically based model for evaluating the photon detection efficiency and the temporal response of SPAD detectors. *Journal of Modern Optics* 58, 3-4 (2011), 210–224. https://doi.org/10.1080/09500340.2010.536590

Heide, F., M. B. Hullin, J. Gregson, and W. Heidrich. 2013. Low-budget Transient Imaging using Photonic Mixer Devices. *ACM Transactions on Graphics (Proc. SIGGRAPH)* 32, 4 (2013), 45:1–45:10.

Heide, F., M. O'Toole, K. Zhang, D. B. Lindell, S. Diamond, and G. Wetzstein. 2017. Robust Non-line-of-sight Imaging with Single Photon Detectors. *CoRR* abs/1711.07134 (2017). arXiv:1711.07134 http://arxiv.org/abs/1711.07134

Heide, F., L. Xiao, W. Heidrich, and M. B. Hullin. 2014. Diffuse Mirrors: 3D Reconstruction from Diffuse Indirect Illumination Using Inexpensive Time-of-Flight Sensors. *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)* (2014).

Hernandez, Q., D. Gutierrez, and A. Jarabo. 2017. A Computational Model of a Single-Photon Avalanche Diode Sensor for Transient Imaging. *arXiv preprint arXiv:1703.02635* (2017).

Jakob, W. 2010. Mitsuba renderer. (2010). http://www.mitsuba-renderer.org.

Jarabo, A., J. Marco, A. Muñoz, R. Buisan, W. Jarosz, and D. Gutierrez. 2014. A framework for transient rendering. *ACM Transactions on Graphics (TOG)* 33, 6 (2014), 177.

Jarabo, A., B. Masia, J. Marco, and D. Gutierrez. 2017. Recent advances in transient imaging: A computer graphics and vision perspective. *Visual Informatics* 1, 1 (2017), 65–79.

Kadambi, A., H. Zhao, B. Shi, and R. Raskar. 2016. Occluded Imaging with Time-of-Flight Sensors. *ACM Transactions on Graphics* 35, 2, Article 15 (March 2016), 12 pages. https://doi.org/10.1145/2836164

Katz, O., P. Heidmann, M. Fink, and S. Gigan. 2014. Non-invasive single-shot imaging through scattering layers and around corners via speckle correlations. *Nature Photonics* 8, 10 (2014), 784–790.

Kirmani, A., T. Hutchison, J. Davis, and R. Raskar. 2009. Looking around the corner using transient imaging. In *Proc. ICCV*. 159–166.

Klein, J., C. Peters, J. Martín, M. Laurenzis, and M. B. Hullin. 2016. Tracking objects outside the line of sight using 2D intensity images. *Scientific Reports* 6, 32491 (2016).

Laurenzis, M. and A. Velten. 2014. Nonline-of-sight laser gated viewing of scattered photons. *Optical Engineering* 53, 2 (2014), 023102–023102.

Levenberg, K. 1944. A Method for the Solution of Certain Non-linear Problems in Least Squares. *Quart. Appl. Math.* 2, 2 (1944), 164–168.

Lorensen, W. and H. Cline. 1987. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. In *Proc. 14th Annual Conf. on Computer Graphics and Interactive Techniques (SIGGRAPH '87)*. ACM, 163–169. https://doi.org/10.1145/37401.37422

Manna, M. L., F. Kine, E. Breitbach, J. Jackson, T. Sultan, and A. Velten. 2018. Error Backprojection Algorithms for Non-Line-of-Sight Imaging. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2018), 1–1. https://doi.org/10.1109/TPAMI.2018.2843363

Marco, J., W. Jarosz, D. Gutierrez, and A. Jarabo. 2017. Transient Photon Beams. In *Spanish Computer Graphics Conference (CEIG)*. The Eurographics Association. https://doi.org/10.2312/ceig.20171216

Marquardt, D. W. 1963. An algorithm for Least-Squares Estimation of Nonlinear Parameters. *SIAM J. Appl. Math.* 11, 2 (1963), 431–441. https://doi.org/10.1137/0111030

Naik, N., S. Zhao, A. Velten, R. Raskar, and K. Bala. 2011. Single view reflectance capture using multiplexed scattering and time-of-flight imaging. *ACM Trans. Graph.* 30, 6 (2011), 171.

O'Toole, M., D. B. Lindell, and G. Wetzstein. 2018a. Confocal non-line-of-sight imaging based on the light-cone transform. *Nature* 555, 25489 (2018), 338–341.

O'Toole, M., D. B. Lindell, and G. Wetzstein. 2018b. Real-time Non-line-of-sight Imaging. In *ACM SIGGRAPH 2018 Emerging Technologies (SIGGRAPH '18)*. ACM, New York, NY, USA, Article 14, 2 pages. https://doi.org/10.1145/3214907.3214920

Pediredla, A. K., M. Buttafava, A. Tosi, O. Cossairt, and A. Veeraraghavan. 2017. Reconstructing rooms using photon echoes: A plane based model and reconstruction algorithm for looking around the corner. In *Computational Photography (ICCP), 2017 IEEE International Conference on*. IEEE, 1–12.

Pharr, M. and G. Humphreys. 2010. *Physically Based Rendering, Second Edition: From Theory To Implementation* (2nd ed.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Schroeder, W., K. Martin, and B. Lorensen. 2006. *The Visualization Toolkit–An Object-Oriented Approach To 3D Graphics* (fourth ed.). Kitware, Inc.

Slaney, M. and P. A. Chou. 2014. *Time of Flight Tracer*. Technical Report. Microsoft Research. https://www.microsoft.com/en-us/research/publication/time-of-flight-tracer/

Smith, A., J. Skorupski, and J. Davis. 2008. *Transient Rendering*. Technical Report UCSC-SOE-08-26. School of Engineering, University of California, Santa Cruz.

Thrampoulidis, C., G. Shulkind, F. Xu, W. T. Freeman, J. H. Shapiro, A. Torralba, F. N. Wong, and G. W. Wornell. 2017. Exploiting Occlusion in Non-Line-of-Sight Active Imaging. *arXiv preprint arXiv:1711.06297* (2017).

Velten, A., R. Raskar, and M. Bawendi. 2011. Picosecond Camera for Time-of-Flight Imaging, In Imaging and Applied Optics. *Imaging and Applied Optics*, IMB4. https://doi.org/10.1364/ISA.2011.IMB4

Velten, A., T. Willwacher, O. Gupta, A. Veeraraghavan, M. Bawendi, and R. Raskar. 2012. Recovering three-dimensional shape around a corner using ultrafast time-of-flight imaging. *Nature Communications* 3 (2012), 745.

Velten, A., D. Wu, A. Jarabo, B. Masia, C. Barsi, C. Joshi, E. Lawson, M. Bawendi, D. Gutierrez, and R. Raskar. 2013. Femto-photography: Capturing and Visualizing the Propagation of Light. *ACM Trans. Graph.* 32, 4, Article 44 (July 2013), 8 pages. https://doi.org/10.1145/2461912.2461928

Wu, D., A. Velten, M. O'Toole, B. Masia, A. Agrawal, Q. Dai, and R. Raskar. 2014. Decomposing Global Light Transport Using Time of Flight Imaging. *International Journal of Computer Vision* 107, 2 (01 Apr 2014), 123–138. https://doi.org/10.1007/s11263-013-0668-2

Wu, D., G. Wetzstein, C. Barsi, T. Willwacher, M. O'Toole, N. Naik, Q. Dai, K. Kutulakos, and R. Raskar. 2012. Frequency analysis of transient light transport with applications in bare sensor imaging. *Computer Vision–ECCV 2012* (2012), 542–555.