Computers & Graphics 83 (2019) 107–113

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Special Section on CEIG

Transient instant radiosity for efficient time-resolved global illumination

Xian Pan*, Victor Arellano, Adrian Jarabo

Universidad de Zaragoza, I3A, Zaragoza, 50018, Spain

ARTICLE INFO

Article history:
Received 1 May 2019
Revised 16 July 2019
Accepted 22 July 2019
Available online 1 August 2019

Keywords:
Transient rendering
Transient instant radiosity

ABSTRACT

Over the last decade, transient imaging has had a major impact in the area of computer graphics and computer vision. The hability of analyzing light propagation at picosecond resolution has enabled a variety of applications such as non-line of sight imaging, vision through turbid media, or visualization of light in motion. However, despite the improvements in capture at such temporal resolution, existing rendering methods are still very time-consuming, requiring a large number of samples to converge to noise-free solutions, therefore limiting the applicability of such simulations. In this work, we generalize instant radiosity, which is very suitable for parallelism in the GPU, to transient state. First, we derive it from the transient path integral, including propagation and scattering delays. Then, we propose an efficient implemention on the GPU, and demonstrate interactive transient rendering with hundreds of thousands of samples per pixel to produce noiseless time-resolved renders.

1. Introduction

In 1964, Prof. Harold Edgerton captured the famous photo *Bullet Through Apple* in the MIT laboratory, where he was able to visualize the movement of a bullet traveling at a speed of approximately 850 m/s, or a obturation time of 4–10 μs. Fifty years later, Velten et al. introduced *femto-photography* [1], which allowed to capture videos at an effective exposure time of 1.85 ps per frame (roughly a trillion frames per second), allowing to capture the propagation of light. The development of this technology has led to a new area in computational imaging, termed *transient imaging* [2]: this field breaks the classical assumption in computer vision and computer graphics of infinite speed of light, allowing the use of such information in numerous applications such as capturing and recognizing materials [3], reconstructing non-line-of-sight (NLOS) geometry [4–6], or imaging through turbid media [7].

Influenced by remarkable advances in the field of *ultra-fast imaging*, and due to the need of reproducing the observed phenomena in order to perform analysis by synthesis, generation of ground truth data in reconstruction techniques, machine learning, etc., simulation of this type of light transport has become more and more important. In 2008, Smith et al. [8] proposed the first framework within the context of the traditional render equation [9], which was later formalized by Jarabo et al. [10], extending the classic *path integral* [11] by adding the temporal domain in the form of propagation and scattering delays.

The use of path integral in rendering techniques such as path tracing has led to realistic results using stochastic integration. However, in order to obtain high quality images, a large number of samples must be processed to reduce stochastic error. Consequently, this results in a high computation time cost. This downside effect is even more critical in the case of transient rendering, where several images need to be generated to visualize how light travels through the scene. Moreover, the spatio-temporal structure of the transient light transport signal makes the traditional sampling and reconstruction techniques [12] unsuitable for this new domain, since most of them generate paths incrementally without any control of the total path length; this lack of control makes very difficult to reconstruct specific frames of an animation, which was alleviated by using different density estimation techniques along the temporal domain [10]. Unfortunately, this still results in the need of generating several thousands of samples to obtain a low-error output. This limits the use of the transient renderer in, for example, inverse rendering [13] or machine learning problems [14], where hundreds or thousands of renders may be necessary to achieve a solution or to have a suitable training dataset.

Therefore, the use of approximate global illumination is needed, which does not aim for exact results, but robust approximations of them in order to reduce computation time. Additionally, since rendering is a very parallelizable process, it is especially suitable to be implemented in the GPU. In this work, we focus on develop-

* Corresponding author.
E-mail address: xpan@iisaragon.es (X. Pan).

https://doi.org/10.1016/j.cag.2019.07.009
0097-8493/

ing a realistic efficient transient renderer, that leverages the GPU computing capacities. We adopt the instant radiosity method [15], that transforms the rendering equation into a many-lights problem, and generalize it to transient state. Then, we demonstrate a GPU-friendly implementation, that allows very efficient renderings of indirect time-resolved global illumination. We are confident that our work will serve in many applications using transient imaging, to serve as a powerful tool for generating datasets for benchmark and training, as well as a forward model for inversion techniques.

2. Related work

The field of physically-based rendering is an extensive domain, with a large number of papers addressing the subject. Here we focus on those most related to our work, and refer the reader to other sources for more details on the field in general [16,17].

Transient rendering. The first to introduce the time of flight of light into the render equation [9] were Smith et al. [8]. Their model provide a solid theory basis for transient rendering, but do not propose a practical framework for the estimation of transient global illumination. Jarabo et al. [10] presented a generalized formulation for light propagation based on the path integral, including scattering and propagation delays in bidirectional path tracing (BDPT). They also proposed sampling techniques adapted to the temporal domain to reduce the resulting variance. Pitts et al. [18] and Adam et al. [19] followed a similar approach, but with a slower convergence rate. Later, Jarabo and Arellano [20] shown that transient light transport can be modeled as a vector-based version of the path integral. Iseringhausen and Hullin [21] proposed an efficient GPU implementation of a two-bounces forward path tracing, in the context of NLOS reconstruction via non-linear optimization. Alternatively, Meister et al. [22,23] used a transient version of photon mapping, resulting in a robust estimation of light transport, and allowing caustic rendering in the transient domain. Ament et al. [24] also used transient photon mapping to solve the refractive radioactive transport equation, where they propose a solution for rendering participating media with different refractive indices. Marco et al. [25] proposed a transient version of the photon beams algorithm for efficient rendering of participating media.

Both BDPT and photon tracing provide robust results when a sufficient number of samples are sampled at the cost of hours and hours of CPU computation. The time-consuming generation of sequential transient videos dramatically limits their use. Several algorithms already exist that try to solve the problem but within a different application domain, mainly in the area of non-line-of-sight imaging, which takes advantage of GPU parallelism to efficiently reconstruct the geometry of occluded objects using the transient state of light: Hullin [26] and Klein et al. [27] demonstrated that a transient image can be approximated using a transient version of the GPU radiosity methods. Our method follows the same lines, to obtain transient videos in an interactive time using GPU-based approximation techniques.

Interactive global illumination. Even though ray tracing techniques (e.g path tracing) can be implemented on GPUs [28–30], complex scenes are generally not suitable for such methods, requiring heavy denoising for noise-less solutions. Instead, using methods based on virtual lights (VPLs)[31], are very suitable for GPU rasterization, do not have this type of limitation, making them widely used for different applications: Keller [15] introduced instant radiosity (IR), which approximates the indirect illumination using a set of virtual point lights placed at the reflecting indirect surfaces. Several extensions were later proposed to improve both the VPL generation process such as bidirectional IR [32], Metropolis IR [33], or to reduce singularities that can be caused by the use of VPLs [34–36]. In order to generate VPLs in real-time, Dachsbacher et al. [37] proposed reflective shadow map (RSM) to use from the light source GPU rasterization to store at each pixel the position, normal and color of the visible object from the camera, so that each pixel represents a VPL. Later, they proposed to splat indirect illumination [38] from the VPLs. To accelerate visibility, Ritschel et al. [39] proposed to approximate the visibility from the shadow maps allowing indirect occlusion at real-time rates. We generalize instant radiosity to resolve transient light transport, using RSM to efficiently generate VPLs in GPUs.

3. Transient instant radiosity

Here we develop a time-resolved generalization for instant radiosity [15]. First we provide an overview of the original method. Then, we develop its time-resolved version by making use of the general path integral formulation.

3.1. Background: Instant Radiosity

The rendering equation [9] models the outgoing radiance $L_o(\mathbf{x}\rightarrow\omega_o)$ at a point $\mathbf{x}$ in a surface in direction ω_o, as

$$L_o(\mathbf{x}\rightarrow\omega_o) = L_e(\mathbf{x}\rightarrow\omega_o) + \int_{H^2} L_i(\mathbf{x}\leftarrow\omega_i)\,\rho(\omega_i\rightarrow\mathbf{x}\rightarrow\omega_o)\cos\theta_i\,d\omega_i, \quad (1)$$

where $L_e(\mathbf{x}\rightarrow\omega_o)$ is the emitted radiance, $L_i(\mathbf{x}\leftarrow\omega_i)$ is the incoming radiance from direction ω_i, $\rho(\omega_i\rightarrow\mathbf{x}\rightarrow\omega_o)$ is the BRDF at $\mathbf{x}$, and $\theta_i =$ is the angle between $\mathbf{n_x}$ the normal at $\mathbf{x}$ and ω_i. Instant radiosity [15] solves Eq. (1) by using a two-pass algorithm, where light particles are traced from the light, and then used to compute the illumination reflected from the surfaces visible from the camera. Such particles model the reflected radiance at their interaction point, acting as *virtual point lights* (VPL). Then, in the rendering pass, the rendering equation at point $\mathbf{x}$ is solved by just computing the contribution of each VPL into x as

$$L_o(\mathbf{x}\rightarrow\omega_o) \approx L_e(\mathbf{x}\rightarrow\omega_o) + \sum_{l=1}^{N} L_o^l(\mathbf{x}\rightarrow\omega_o), \quad (2)$$

with

$$L_o^l(\mathbf{x}\rightarrow\omega_o) = \frac{L_e^l(\mathbf{x}_l\rightarrow\mathbf{x})V(\mathbf{x}_l\leftrightarrow\mathbf{x})}{|\mathbf{x}-\mathbf{x}_l|^2}\rho(\omega_i\rightarrow\mathbf{x}\rightarrow\omega_o)\cos\theta_i, \quad (3)$$

where N is the number of VPLs in the scene, $L_e^l(\mathbf{x}_l\rightarrow\mathbf{x})$ is the radiance emited by the VPL l placed at $\mathbf{x}_l$ towards $\mathbf{x}$, and $V(\mathbf{x}_l\leftrightarrow\mathbf{x})$ is the binary visibility function between $\mathbf{x}$ and $\mathbf{x}_l$.

3.2. Transient instant radiosity

In order to define our time-resolved generalization of VPL-based global illumination, we will first generalize the rendering equation [Eq. (1)] to transient state, by including the temporal dependence as

$$L_o(\mathbf{x}\rightarrow\omega_o,t) = L_e(\mathbf{x}\rightarrow\omega_o,t) + L_r(\mathbf{x}\rightarrow\omega_o,t), \quad (4)$$

$$L_r(\mathbf{x}\rightarrow\omega_o,t) = \int_{H^2}\int_{t_0}^{t} L_i(\mathbf{x}\leftarrow\omega_i,t')\,\rho(\omega_i\rightarrow\mathbf{x}\rightarrow\omega_o,t-t')\cos\theta_i\,dt'\,d\omega_i, \quad (5)$$

where we include the temporal emission profile in the emission $L_e(\mathbf{x}\rightarrow\omega_o,\,t)$, as well as the scattering delays encoded in $\rho(\omega_i\rightarrow\mathbf{x}\rightarrow\omega_o,t-t')$, which can be due to e.g. multiple internal reflections within micro-geometry [40,41] or inelastic scattering effects

such as fluorescence [42]. Last, t_0 is an initial time usually set to $t_0 = -\infty$. To solve Eq. (4), we can define our transient instant radiosity by accounting for the temporal domain in Eqs. (2) and (3) as

$$L_o(\mathbf{x} \to \omega_o, t) \approx L_e(\mathbf{x} \to \omega_o, t) + \sum_{l=1}^{N} L_r^l(\mathbf{x} \to \omega_o, t), \tag{6}$$

$$L_r^l(\mathbf{x} \to \omega_o, t) = \frac{\cos\theta_i V(\mathbf{x}_l \leftrightarrow \mathbf{x})}{|\mathbf{x} - \mathbf{x}_l|^2} \times \int_{t_0}^{t} L_e^l(\mathbf{x}_l \to \mathbf{x}, t_l(t) - t') \rho(\omega_i \to \mathbf{x} \to \omega_o, t - t')\, dt', \tag{7}$$

where $t_l(t) = t - t(\mathbf{x} \leftrightarrow \mathbf{x}_l)$ accounts for the propagation delay due to light travelling from the light source to $\mathbf{x}$ as

$$t(\mathbf{x}_x \leftrightarrow \mathbf{x}_y) = \int_{s_x}^{s_y} \frac{\eta(\mathbf{x}_r)}{c}\, dr, \tag{8}$$

where r parametrizes the path of light between the two points, s_x and s_y are the parameters of the path at $\mathbf{x}_x$ and $\mathbf{x}_y$, respectively, c is the speed of light in vacuum and $\eta(\mathbf{x}_r)$ represents the index of refraction of the medium at $\mathbf{x}_r$. In order to solve Eqs. (6) and (7) we therefore need to define the time-resolved emitting function $L_e^l(\mathbf{x} \to \omega_o, t)$. To do so, we will make use of the path integral view of instant radiosity.

Transient Instant Radiosity as a Path Integral Estimator. In order to define our time-resolved generalization of VPL-based global illumination, we will work within the path integral [11] framework. As shown by Walter et al. [43] and Krivanek et al. [44], instant radiosity can be understood as a bidirectional path tracer where the light paths are reused along all eye paths. While this generates some artifacts due to correlation, it ensures relatively small variance in smooth areas. Let us start with the transient path integral proposed by Jarabo et al. [10], which defines the light contributing in a (time-resolved) pixel I as the integral over the all light transport paths Ω arriving to this pixel as

$$I = \int_{\Omega} \int_{\Delta T} f(\bar{\mathbf{x}}, \overline{\Delta \mathbf{t}})\, d\mu(\overline{\Delta \mathbf{t}})\, d\mu(\bar{\mathbf{x}}), \tag{9}$$

where ΔT is the space of temporal delays of all paths, $\bar{\mathbf{x}} = \mathbf{x}_0 \dots \mathbf{x}_k$ is the full light path of length k, consisting of $k+1$ vertices, where vertices $\mathbf{x}_0$ and $\mathbf{x}_k$ lie on a light source and camera sensor, respectively, and $\mathbf{x}_1 \dots \mathbf{x}_{k-1}$ are intermediate scattering vertices. The differential measure $d\mu(\bar{\mathbf{x}})$ is the area integration for surfaces vertices. Finally, $\overline{\Delta \mathbf{t}} = \Delta t_0 \dots \Delta t_k$ represents the sequence of scattering delays in $\rho(\omega_i \to \mathbf{x} \to \omega_o, t - t')$ along the path, and $d\mu(\overline{\Delta \mathbf{t}})$ is the temporal integration measure at each path vertex. The path contribution $f(\bar{\mathbf{x}}, \overline{\Delta \mathbf{t}})$ is defined as

$$f(\bar{\mathbf{x}}, \overline{\Delta \mathbf{t}}) = L_e(\mathbf{x}_0 \to \mathbf{x}_1, \Delta t_0)\, \mathfrak{T}(\bar{\mathbf{x}}, \overline{\Delta \mathbf{t}})\, W_e(\mathbf{x}_{k-1} \to \mathbf{x}_k, \Delta t_k), \tag{10}$$

where L_e, W_e and $\mathfrak{T}$ are the time-dependent emission, sensor importance, and path throughput, respectively. The sensor importance W_e defines the spatial and angular sensitivity as in the steady-state path integral, but also the region of time we are interested in evaluating (e.g. in a streak-camera or SPAD sensor, the temporal dependence of W_e for a pixel-frame would be the temporal bucket of that particular frame). The transient path throughput is defined as:

$$\mathfrak{T}(\bar{\mathbf{x}}, \overline{\Delta \mathbf{t}}) = \left[\prod_{i=1}^{k-1} \rho(\mathbf{x}_i, \Delta t_i)\right] \left[\prod_{i=0}^{k-1} G(\mathbf{x}_i, \mathbf{x}_{i+1}) V(\mathbf{x}_i, \mathbf{x}_{i+1})\right]. \tag{11}$$

where $\rho(\mathbf{x}_i, \Delta t_i)$ is the scattering event at $\mathbf{x}_i$, $G(\mathbf{x}_i, \mathbf{x}_{i+1}) = \cos\theta_i \cos\theta_{i+1} |\mathbf{x}_i - \mathbf{x}_{i+1}|^{-2}$ is the geometric term, and $V(\mathbf{x}_i, \mathbf{x}_{i+1})$ is the binary visibility between $\mathbf{x}_i$ and $\mathbf{x}_{i+1}$. Note that Eq. (11) is essentially the same as in the classic path integral, with the exception that the scattering operator $\rho(\mathbf{x}_i, \Delta t_i)$ includes a temporal dependence modeling the scattering delay described before.

In order to fully define a transient light path, we need to account for the spatial, and temporal coordinates. The temporal coordinate at each path vertex $\mathbf{x}_i$ are t_i^-, computed as

$$t_i^- = \sum_{j=0}^{i-1} \left(t(\mathbf{x}_j \leftrightarrow \mathbf{x}_{j+1}) + \Delta t_j\right), \quad t_i = t_i^- + \Delta t_i, \tag{12}$$

where t_0 and t_k are the emission and detection times of a light path, and $t(\mathbf{x}_j \leftrightarrow \mathbf{x}_{j+1})$ is the propagation delay between vertices $\mathbf{x}_j$ and $\mathbf{x}_{j+1}$ defined following Eq. (8).

The transient path integral (9) can be numerically approximated using a Monte Carlo estimator:

$$\langle I \rangle = \frac{1}{M} \sum_{j=1}^{M} \frac{f(\bar{\mathbf{x}}_j, \overline{\Delta \mathbf{t}}_j)}{p(\bar{\mathbf{x}}_j, \overline{\Delta \mathbf{t}}_j)}, \tag{13}$$

which averages M random paths $\bar{\mathbf{x}}_j, \overline{\Delta \mathbf{t}}_j$ drawn from a spatio-temporal probability distribution (pdf) $p(\bar{\mathbf{x}}_i, \overline{\Delta \mathbf{t}}_i)$ defined by the chosen path and time delay sampling strategy. Similar to bidirectional pathtracing, methods based on VPLs compute Eq. (13) by sampling a set of subpaths from the light source (called light subpaths) $\bar{\mathbf{x}}_l = \mathbf{x}_0 \dots \mathbf{x}_{k_l}$ of length $k_l \in [1, k-1]$, and connecting them with the subpaths generated from the camera (eye subpaths) $\bar{\mathbf{x}}_w = \mathbf{x}_{k_l+1} \dots \mathbf{x}_k$ via deterministic shadow connections, to generate a full path $\bar{\mathbf{x}} = [\bar{\mathbf{x}}_l, \bar{\mathbf{x}}_w] = \mathbf{x}_0 \dots \mathbf{x}_{k_l} \mathbf{x}_{k_l+1} \dots \mathbf{x}_k$, where $[\cdot]$ is the concatenator operator. The main difference with respect to bidirectional path tracing is that the light subpaths are reused for all eye subpaths in the image, so that $\langle I \rangle$ becomes

$$\langle I \rangle = \frac{1}{MN} \sum_{j=1}^{M} \frac{f(\bar{\mathbf{x}}_w^j, \overline{\Delta \mathbf{t}}_j)}{p(\bar{\mathbf{x}}_w^j, \overline{\Delta \mathbf{t}}_j)} \sum_{l=1}^{N} \times \left(\frac{f(\bar{\mathbf{x}}_l^l, \overline{\Delta \mathbf{t}}_l)}{p(\bar{\mathbf{x}}_l^l, \overline{\Delta \mathbf{t}}_l)} \rho(\mathbf{x}_{k_l}, \Delta t_{k_l}) G(\mathbf{x}_{k_l}, \mathbf{x}_{k_l+1}) \rho(\mathbf{x}_{k_l+1}, \Delta t_{k_l+1}) V(\mathbf{x}_{k_l}, \mathbf{x}_{k_l+1})\right) \tag{14}$$

where $f(\bar{\mathbf{x}}_w^j, \overline{\Delta \mathbf{t}}_j) = \mathfrak{T}(\bar{\mathbf{x}}_w^j, \overline{\Delta \mathbf{t}}^j) W_e(\mathbf{x}_{k-1} \to \mathbf{x}_k, \Delta t_k)$ and $f(\bar{\mathbf{x}}_l^l, \overline{\Delta \mathbf{t}}_l) = L_e(\mathbf{x}_0 \to \mathbf{x}_1, \Delta t_0) \mathfrak{T}(\bar{\mathbf{x}}_l^l, \overline{\Delta \mathbf{t}}^l)$ are measurement of the eye and light subpaths, $\bar{\mathbf{x}}_w^j$ and $\bar{\mathbf{x}}_l^l$ respectively, M is the number of eye subpaths in a pixel, and N is the number of VPLs generated.

Eq. (14) implicitly relates with Eqs. (6) and (7) by defining the emission term $L_e^l(\mathbf{x} \to \omega_o, t)$ as

$$L_e^l(\mathbf{x} \to \omega_o, t) = \frac{1}{N} \delta(t - t_l) \frac{f(\bar{\mathbf{x}}_l^l, \overline{\Delta \mathbf{t}}_l)}{p(\bar{\mathbf{x}}_l^l, \overline{\Delta \mathbf{t}}_l)} \rho(\mathbf{x}_{k_l}, \Delta t_{k_l}) \cos\theta_{k_l}, \tag{15}$$

where $\delta(t - t_l)$ is the Dirac delta function modeling the impulse emission of the VPL, where t_l is the emisison time of the VPL computed using Eq. (12). Note the integration along the temporal domain in Eq. (7) is implicitly solved numerically by the sampled variable $\overline{\Delta \mathbf{t}}$. In essence, in the first pass of the algorithm we generate a set of VPLs by sampling a set of light subpaths; each VPL would have a spatio-temporal emitting function defined by Eq. (15). Then, in render time we simply evaluate Eqs. (6) and (7) using Eq. (15).

4. Implementation

We implemented our transient instant radiosity renderer using OpenGL and Java. In order to fit it into the raster pipeline of the GPU, we perform a set of simplifications. First, we assume

that only one- and two-segment paths exist in the scene. Taking Eq. (9) and explicitly separating bounces, we get

$$I = \sum_{b=1}^{\infty} \int_{\Omega^b} \int_{\Delta T^b} f(\overline{\mathbf{x}}, \overline{\Delta \mathbf{t}})\, \mathrm{d}\mu(\overline{\Delta \mathbf{t}})\, \mathrm{d}\mu(\overline{\mathbf{x}}), \tag{16}$$

where Ω^b and ΔT^b are the space of paths and temporal delays of paths of b segments. By clamping the number of segments to $b = 3$ we end up having direct and one-indirect-bounce lighting, as well as light incoming directly from the light source to the camera. This results into an energy loss, but according to the study of Tabellion and Lamorlette [45], a single indirect light bounce is enough to obtain plausible results in many cases. Note however that this claim is based on steady-state images; in transient state it would mean that longer path lengths (i.e. light arriving at longer paths) will be lost. However, in most practical applications [2] most information is encoded in the first and second-order scattering. The second approximation consists into assuming that all scattering delays are neglectable, and that can be approximated by a delta function. This allows us to trivially solve the integrand along the temporal domain. Finally, we assume all surfaces to be perfectly diffuse. These last two approximations do not introduce bias in the final result but limit the transient phenomena that our implementation can simulate.

For rendering, we use the two classical passes of instant radiosty (VPL generation, and rendering), and introduce a last step for reconstructing the temporal dimension of the scene. For this last step, following the terminology proposed by Jarabo et al. [10], we use the histogram density estimation for reconstruction, due to its simplicity to work with delta functions. More advanced reconstruction approaches are left as future work.

Generating the VPLs. In order to generate the VPLs on the GPU, we rely on reflective shadow maps (RSM) [37]: we render the scene from the point of view of the light source, storing the surface albedo $\alpha(p_l)$, normal $\mathbf{n}_\mathbf{x}(p_l)$, and distances from the camera $d(p_l)$ for each pixel p_l in the RSM. With these parameters, and leveraging the assumption of two-bounces light transport and diffuse reflectance, we can define Eq. (15) for each VPL as

$$L_\mathrm{e}^l(\mathbf{x} \to \omega_\mathrm{o}, t) \approx \delta(t - t_l) \frac{L(p_l)}{\Omega(p_l)} \frac{\alpha(p_l)}{\pi} \cos\theta_{k_l}, \tag{17}$$

where $t_l = d(p)\eta/c$, and $\Omega(p)$ is the solid angle of pixel p, which decreases quadrately with the number of VPLs N, and $L(p)$ is emitting function of the light source, which in our case is a spotlight with constant emission along the light frustum. After generating the RSM values in the first pass of the render, which are four independent textures, these are used in the final pass to calculate the global illumination in the rendering pass.

Rendering. In the second render pass, we compute Eq. (6) from the VPLs generated in the first pass, by using a deferred rendering approach. In order to do so, we need to compute the visibility function $V(\mathbf{x}, \mathbf{x}_l)$ between the shaded point $\mathbf{x}$ and the VPL. In order to compute it efficiently, we use an approximated visibility based on imperfect shadow maps (ISM) [39], where a large number of hemispherical shadow maps are generated for each VPL in parallel by using a coarse point-based approximation of the scene.

The use of VPLs provides good results for the calculation of global illumination, but introduces certain errors that are easily perceptible to the human eye when the number of VPLs used is low. One of them is the existence of lighting peaks in the scene due to the singularity produced by the inverse squared distance of the geometric term. To remove artifacts due to this singularity, we clamp the geometric term to a user specified value. We opt for this approach since for diffuse scenes the energy lost introduced by clamping does not result into significant appearance changes [46]. Note however that this results in an energy lost in the scene, generating dark areas especially in cavities and corners.

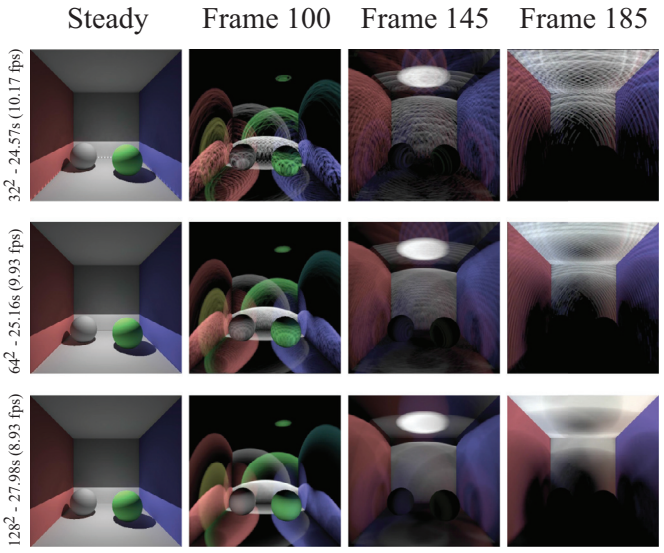

Fig. 1. Cornell Box results with the same ISM size (128^2) and point samples per VPL (2000), but with different RSM sizes (i.e. increasing number of VPLs). As we increase the number of VPLs, the noise in the rendered animation vanishes. Given the deterministic location of the VPLs along a regular grid, for small RSM resolution the banding produced in the temporal domain is very noticeable; for future work it would be interesting to incorporate higher-quality reconstruction to remove such banding for low VPL counts. Note that for all VPL counts, the results in steady state look very similar.

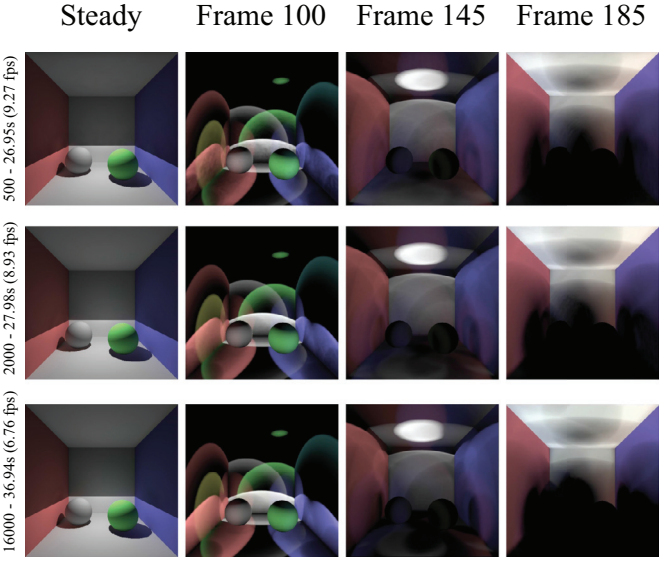

Fig. 2. Cornell Box results with same ISM size (128^2) and RSM size (128^2), but with different point samples per VPL. With a small number of samples used for ISM, it is possible to generate an ISM with many holes, so as can be seen in frame 145, the indirect light shadow is not generated correctly for those cases. However, this error is not very significant, and we can ignore it in exchange for faster computation time.

Generating the space-time volumes. Finally, once the space-time radiance samples are simulated, we need to use them to reconstruct the transient video. Unlike a stationary render, which uses a 2D texture to store the result (for each point, saves the sum of all radiance), in a transient renderer we save the data in a 3D texture. For that, we use an approach based on histogram density estima-

Algorithm 1 Algorithm pseudocode.

```
rsm = RSM() // Generate RSM with the VPLs
points = GenerateISMPoints() // Generate points on surfaces for building ISMs
for all v in batching video do
    vs = VideoSegment(v)
    DirectBounce(vs, rsm)
    for all i in batching ISM do
        ism = ISM(i, points) // Compute ISMs for VPLs in batch i
        IndirectBounce(vs, ism, rsm) // Compute indirect illumination from VPLs in i
        ClearSegment(ism) // Clean ISM of batch i from memory
    end for
    GPUtoCPU(vs) // Transfer space-time volume for batch v
    ClearSegment(vs) // Clean space-time volume of batch v from memory
end for
```

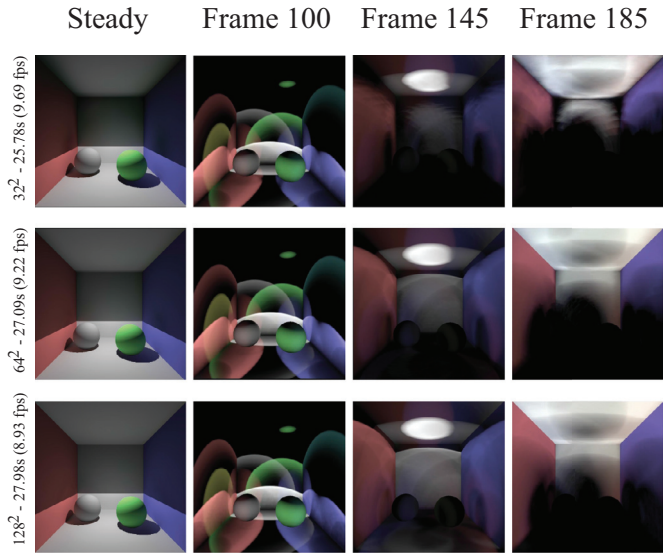

Fig. 3. Cornell Box results with same RSM size (128^2) and point samples per VPL (2000), but with different ISM sizes. It has the same problem as the RSM size, but only affects the visibility of indirect lights.

tion: For each radiance sample in a point, we assign it to the corresponding cell depending on the light propagation time from the source to that point (going through VPLs or not) calculated with Eqs. (8) and (12). We follow an approach similar to the one used for GPU voxelization [47], where each spatio-temporal sample is rasterized into the space-time volume.

4.1. Discussion

Our algorithm is bounded by the memory requirements imposing by storing the results in a large 3D texture, as well as the auxiliary buffers needed for the VPLs computation. To store an HDR color video with resolution X and Y and duration t (in frames) we need $16 \times X \times Y \times t$ bytes. For example, for a video with resolution 1024×1024 and 250 frames, we need 4GB approximately. The auxiliary buffers of the VPLs, mainly the RSM and the ISM additionally require a storage of order $\mathcal{O}(N \times X_s \times Y_s)$, with N the number of VPLs, and X_s, Y_s the resolution of each shadow map.

As a result, it is not possible to store all the information in the GPU at the same time when GPU memory is insufficient. We address that by processing the VPLs in batches, so that a subset of VPLs are generated, processed, and removed from memory before the next one is, and then the next one is processed. We have made two segmentations: one for the ISM (batching the VPLs as they are processed) and the final transient volume (splitting the video along the spatial domain). Algorithm 1 shows a pseudocode of the implementation: We iterate on the different batches of the transient volume, calculate the corresponding direct and indirect lighting in GPU for each batch, and then transfer it back to the main memory, freeing space for the results of the next batch. We do not store any ISM on the main memory; while this means that we do not amortize the cost of computing them between batches, we found that the time to regenerate them in GPU is much faster than the memory transfer that would involve a CPU-to-GPU load.

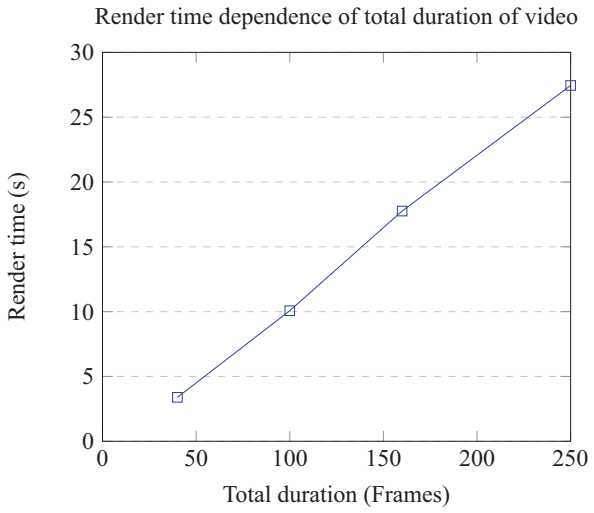

Fig. 4. Dependency of the render time of the video duration in the Cornell Box scene with 128^2 RSM size, 64^2 ISM size and 4000 ISM samples. The render time is proportional to the video duration. This is mainly due to the fact that the size of the 3D texture needed to store the video is larger when the video duration is longer, so that the time needed to transfer data from GPU to CPU is longer.

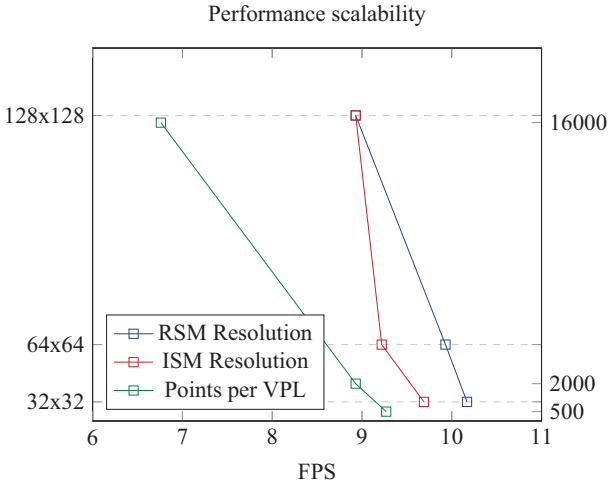

Fig. 5. Performance scalability of the Cornell Box scene with respect to the RSM resolution, ISM resolution and point samples per VPL. For each line, only the indicated parameter varies. The base dimension is 128^2 RSM size, 128^2 ISM size and 2000 point samples per VPL.

5. Results

In the following, we present the results provided by our implementation, generated on worlkstation with a 2.8 GHz CPU and a NVIDIA GeForce 1060. The results are rendered with 1024 × 1024 camera resolution and 250 frames of duration (around 10 s). All the render times specified below refer to GPU render time plus GPU-to-CPU data transfer time. Videos of the shown results are included as supplementary material.

Fig. 6. TABLE scene rendered in 131.76s (1.9 fps) with 256^2 RSM and ISM size, and 24,000 point samples per VPL. The first one is the steady image, and the rest from left to right are fragments of transient video ordered chronologically. We can see the indirect light propagating and leaving the different shadows.

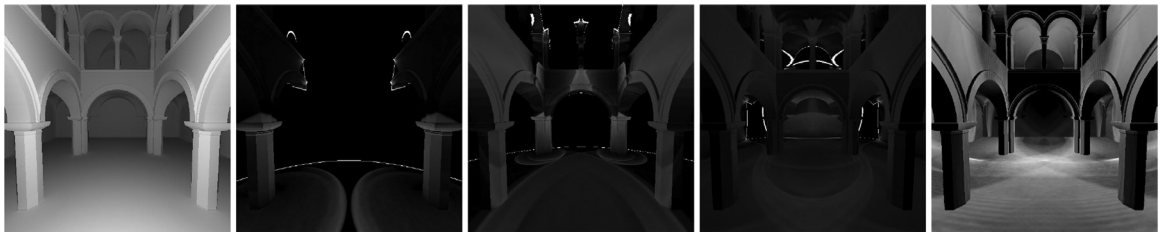

Fig. 7. SPONZA rendered with 512^2 RSM size, 256^2 ISM size and 70,000 point samples per VPL. It takes 438.56 seconds to create a 25 s (600 frames) video with the best quality. The light comes out of the camera and travels through the whole scene. (The last one is brighter because of the tonemapper.).

The selection of parameters for our methods has a big influence on both the final quality and the computation time. These parameters are: number of VPLs, number of points per VPL and ISM resolution (the shadow map resolution for each VPL). Figs. 1–3 compare the effect of these parameters and their costs. A logarithmic tone-mapping has been used for visualization. Frames 100, 145 and 185 have been chosen for the comparison.

Fig. 4 shows the dependence of the render time on the temporal resolution. This parameter has no effect on the GPU render time given the histogram method used. However, it affects the transfer time of the GPU-to-CPU data due to texture size variation.

Fig. 5 shows the comparison of the algorithm performance when varying the different parameters. We can observe that as the dimension of the parameters increases, the fps decreases. The most significant change occurs with the number of points per VPLs used to generate the ISM, because this number together with the number of VPLs equals the total number of points that it is necessary to pass to GPU to compute the ISM.

The selection of the VPL count is quite important: A small number of VPLs can cause artifacts due to banding on the signal, while a large number of VPLs increases the computational cost of the algorithm.

Finally, setting the the number of points per VPL and ISM resolution is closely related, and the quality also depends on the complexity of the scenes and the geometry of the objects. In simple scenes a low number of both parameters can be used, whereas when the complexity of the scene grows (number of objects, geometry complexity, scene size), more points are needed to correctly define the scene, increasing the rendering time as shown in Figs. 6 and 7.

6. Conclusions and future work

In this work we have presented a method to efficiently render scenes in transient state: we have reformulated the light transport in transient state using instant radiosity, so that it is possible to solve the calculation in interactive time. And we have implemented this calculation in GPU taking advantage of its parallelism, using techniques such as reflective shadow map and imperfect shadow map. We believe that efficient transient rendering on the GPU will enable novel transient-based applications based on e.g. inverse rendering for optimization or machine learning. In that context, our technique will be very useful to create novel databases using transient rendering (e.g. [48]).

In the future, we intend to reduce the number of VPLs used so that we can improve computational time, using more sophisticated and modern algorithms. In addition, our renderer offers a first approximation of transient light transport in GPU applying different assumptions, so it only shows limited effects. One of the future steps is to eliminate these assumptions by incorporating glossy materials, or participating media. Finally, efficiently computing the radiance derivatives to increase the applicability of the method in optimization and learning applications is an interesting avenue of future work.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We want to thank the reviewers for their insightful comments. This project has been funded by DARPA (project REVEAL), the European Research Council (ERC) under the EU's Horizon 2020 research and innovation programme (project CHAMELEON, Grant no. 682080), the Spanish Ministry of Economy and Competitiveness (project TIN2016-78753-P), and the BBVA Foundation through a Leonardo grant. Production baby: Noa Jarabo Martínez.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.cag.2019.07.009

References

[1] Velten A, Wu D, Jarabo A, Masia B, Barsi C, Joshi C, et al. Femto-photography: capturing and visualizing the propagation of light. ACM Trans Graph (SIGGRAPH 2013) 2013;32(4).
[2] Jarabo A, Masia B, Marco J, Gutierrez D. Recent advances in transient imaging: a computer graphics and vision perspective. Vis Inf 2017;1(1).
[3] Naik N, Zhao S, Velten A, Raskar R, Bala K. Single view reflectance capture using multiplexed scattering and time-of-flight imaging. ACM Trans Graph (ToG) 2011;30:171.

[4] Velten A, Willwacher T, Gupta O, Veeraraghavan A, Bawendi MG, Raskar R. Recovering three-dimensional shape around a corner using ultrafast time-of-flight imaging. Nat Commun 2012;3:745.
[5] O'Toole M, Lindell DB, Wetzstein G. Confocal non-line-of-sight imaging based on the light-cone transform. Nature 2018;555(7696):338.
[6] Liu X, Guillén I, La Manna M, Nam JH, Reza SA, Le TH, et al. Phasor fields: virtual wave optics for non-line-of-sight imaging. Nature 2019.
[7] Wu R, Jarabo A, Suo J, Dai F, Zhang Y, Dai Q, et al. Adaptive polarization-difference transient imaging for depth estimation in scattering media. Opt Lett 2018;43(6).
[8] Smith A, Skorupski J, Davis J. Transient rendering. Tech. Rep.; 2008.
[9] Kajiya JT. The rendering equation. SIGGRAPH Comput Graph 1986;20(4):143–50. doi:10.1145/15886.15902.
[10] Jarabo A, Marco J, Muñoz A, Buisan R, Jarosz W, Gutierrez D. A framework for transient rendering. ACM Trans Graph (SIGGRAPH Asia 2014) 2014;33(6).
[11] Veach E. Robust Monte Carlo methods for light transport simulation no. 1610. Stanford University; 1997. (Ph.D thesis)
[12] Zwicker M, Jarosz W, Lehtinen J, Moon B, Ramamoorthi R, Rousselle F, et al. Recent advances in adaptive sampling and reconstruction for monte carlo rendering. In: Computer graphics forum, 34. Wiley Online Library; 2015. p. 667–81.
[13] Seitz SM, Matsushita Y, Kutulakos KN. A theory of inverse light transport. In: Proceedings of the tenth IEEE international conference on computer vision, 2005. ICCV 2005, 2. IEEE; 2005. p. 1440–7.
[14] Marco J, Hernandez Q, Muñoz A, Dong Y, Jarabo A, Kim M, et al. Deeptof: off-the-shelf real-time correction of multipath interference in time-of-flight imaging. ACM Trans. Graph. (SIGGRAPH Asia 2017) 2017;36(6).
[15] Keller A. Instant radiosity. In: Proceedings of the twenty-forth annual conference on computer graphics and interactive techniques. SIGGRAPH '97. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co.; 1997. p. 49–56. ISBN 0-89791-896-7. doi:10.1145/258734.258769.
[16] Pharr M, Jakob W, Humphreys G. Physically based rendering: from theory to implementation. 3rd. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.; 2016. ISBN 0128006455, 9780128006450
[17] Ritschel T, Dachsbacher C, Grosch T, Kautz J. The state of the art in interactive global illumination. In: Computer graphics forum, 31. Wiley Online Library; 2012. p. 160–88.
[18] Pitts P, Benedetti A, Slaney M, Chou P. Time of flight tracer. Technical Report 2014.
[19] Adam A, Dann C, Yair O, Mazor S, Nowozin S. Bayesian time-of-flight for realtime shape, illumination and albedo. IEEE Trans. Pattern Anal. Mach. Intell. 2017;39(5):851–64. doi:10.1109/TPAMI.2016.2567379.
[20] Jarabo A, Arellano V. Bidirectional rendering of vector light transport. Comput. Graph. Forum 2018;37(6).
[21] Iseringhausen J, Hullin MB. Non-line-of-sight reconstruction using efficient transient rendering. arXiv:1809080442018.
[22] Meister S, Nair R, Jähne B, Kondermann D. Photon mapping based simulation of multi-path reflection artifacts in time-of-flight sensors. Technical Report. Heidelberg Collaboratory for Image Processing, 2013. 2; 2013.
[23] Meister S, Nair R, Kondermann D. Simulation of time-of-flight sensors using global illumination 2013.
[24] Ament M, Bergmann C, Weiskopf D. Refractive radiative transfer equation. ACM Trans Graph (TOG) 2014;33(2):17.
[25] Marco J, Guillén I, Jarosz W, Gutierrez D, Jarabo A. Progressive transient photon beams. Comput Graph Forum 2019.
[26] Hullin MB. Computational imaging of light in flight. In: Optoelectronic imaging and multimedia technology III, 9273. International Society for Optics and Photonics; 2014. p. 927314.
[27] Klein J, Peters C, Martín J, Laurenzis M, Hullin MB. Tracking objects outside the line of sight using 2d intensity images. Sci Rep 2016;6:32491.
[28] Wald I, Kollig T, Benthin C, Keller A, Slusallek P. Interactive global illumination using fast ray tracing. In: Proceedings of the thirteenth eurographics workshop on rendering. EGRW '02. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association; 2002. p. 15–24. ISBN 1-58113-534-3
[29] Parker SG, Friedrich H, Luebke D, Morley K, Bigler J, Hoberock J, et al. Gpu ray tracing. Commun ACM 2013;56(5):93–101.
[30] Heitz E, Hill S, McGuire M. Combining analytic direct illumination and stochastic shadows. In: Proceedings of the ACM SIGGRAPH symposium on interactive 3D graphics and games. ACM; 2018. p. 2.
[31] Dachsbacher C, Křivánek J, Hašan M, Arbree A, Walter B, Novák J. Scalable realistic rendering with many-light methods. Comput Graph Forum 2014;33(1):88–104. doi:10.1111/cgf.12256.
[32] Segovia B, Iehl JC, Mitanchey R, Proche B. Bidirectional instant radiosity. In: Proceedings of the seventeenth eurographics workshop on rendering; 2006. p. 389–98.
[33] Segovia B, Iehl J, Peroche B. Metropolis instant radiosity. Comput Graph Forum 2007. doi:10.1111/j.1467-8659.2007.01065.x.
[34] Hašan M, Křivánek J, Walter B, Bala K. Virtual spherical lights for many-light rendering of glossy scenes. ACM Trans Graph (TOG) 2009;28:143.
[35] Kollig T, Keller A. Illumination in the presence of weak singularities. In: Monte Carlo and quasi-Monte Carlo methods 2004. Springer; 2006. p. 245–57.
[36] Engelhardt T, Novák J, Schmidt T-W, Dachsbacher C. Approximate bias compensation for rendering scenes with heterogeneous participating media. In: Computer Graphics Forum, 31. Wiley Online Library; 2012. p. 2145–54.
[37] Dachsbacher C, Stamminger M. Reflective shadow maps. In: Proceedings of the 2005 symposium on interactive 3D graphics and games. ACM; 2005. p. 203–31.
[38] Dachsbacher C, Stamminger M. Splatting indirect illumination. In: Proceedings of the 2006 symposium on interactive 3D graphics and games. ACM; 2006. p. 93–100.
[39] Ritschel T, Grosch T, Kim MH, Seidel H-P, Dachsbacher C, Kautz J. Imperfect shadow maps for efficient computation of indirect illumination. ACM Trans Graph 2008;27(5):129:1–129:8. doi:10.1145/1409060.1409082.
[40] Heitz E, Hanika J, d'Eon E, Dachsbacher C. Multiple-scattering microfacet BSDFS with the smith model. ACM Trans Graph 2016;35(4):58:1–58:14. doi:10.1145/2897824.2925943.
[41] Lee JH, Jarabo A, Jeong DS, Gutierrez D, Kim MH. Practical multiple scattering for rough surfaces. ACM Trans Graph (SIGGRAPH Asia 2018) 2018;37(6).
[42] Wilkie A, Tobler RF, Purgathofer W. Combined rendering of polarization and fluorescence effects. Tech. Rep. TR-186-2-01-11. Favoritenstrasse 9–11/186, A-1040 Vienna, Austria: Institute of Computer Graphics and Algorithms, Vienna University of Technology; 2001. Human contact: technical-report@cg.tuwien.ac.at; https://www.cg.tuwien.ac.at/research/publications/2001/wilkie-2001-crp/.
[43] Křivánek J, Hašan M, Arbree A, Dachsbacher C, Keller A, Walter B. Optimizing realistic rendering with many-light methods. In: ACM SIGGRAPH 2012 Courses. SIGGRAPH '12. New York, NY, USA: ACM; 2012. p. 7:1–7:217. ISBN 978-1-4503-1678-1. doi:10.1145/2343483.2343490.
[44] Křivánek J, Georgiev I, Kaplanyan A, Canada J. Path integral methods for light transport simulation: theory & practice. In: EUROGRAPHICS 2014 - Tutorials. Eurographics Association; 2014.
[45] Tabellion E, Lamorlette A. An approximate global illumination system for computer generated films. In: ACM SIGGRAPH 2008 Classes. SIGGRAPH '08. New York, NY, USA: ACM; 2008. p. 74:1–74:8. doi:10.1145/1401132.1401227.
[46] Křivánek J, Ferwerda JA, Bala K. Effects of global illumination approximations on material appearance. ACM Trans Graph 2010;29(4):112:1–112:10. doi:10.1145/1778765.1778849.
[47] Arellano V, Gutierrez D, Jarabo A. Fast back-projection for non-line of sight reconstruction. Opt Express 2017;to appear.
[48] Galindo M, Marco J, OToole M, Wetzstein G, Gutierrez D, Jarabo A. A dataset for benchmarking time-resolved non-line-of-sight imaging. In: Proceedings of ICCP 2019, Posters; 2019.